Parallels Cloud Server 6.0

User's Guide

Copyright © 1999-2012 Parallels IP Holdings GmbH and its affiliates. All rights reserved.

Parallels IP Holdings GmbH.
Vordergasse 59
CH8200 Schaffhausen
Switzerland
Tel: + 41 526320 411
Fax: + 41 52672 2010
www.parallels.com

Copyright © 1999-2012 Parallels IP Holdings GmbH and its affiliates. All rights reserved.

This product is protected by United States and international copyright laws. The product's underlying technology, patents, and trademarks are listed at http://www.parallels.com/trademarks.

Microsoft, Windows, Windows Server, Windows NT, Windows Vista, and MS-DOS are registered trademarks of Microsoft Corporation.
Apple, Mac, the Mac logo, Mac OS, iPad, iPhone, iPod touch, FaceTime HD camera and iSight are trademarks of Apple Inc., registered in the US and other countries.
Linux is a registered trademark of Linus Torvalds.
All other marks and names mentioned herein may be trademarks of their respective owners.

Contents

Introduction 10
About This Guide 10
Organization of This Guide 10
Documentation Conventions 11
Getting Help 12
Feedback 12
Learning Parallels Cloud Server 6.0 Basics 14
Parallels Cloud Server 6.0 Overview 14
OS Virtualization Layer 16
Basics of OS Virtualization 16
Parallels Containers 16
Memory and IOPS Deduplication 18
Templates 18
Hardware Virtualization Layer 19
Basics of Hardware Virtualization 19
Parallels Virtual Machines 19
Virtual Machine Hardware 20
Virtual Machine Files 21
Support of Virtual and Real Media 21
Parallels Cloud Server Configuration 22
Resource Management 23
Understanding Licensing 23
Physical Server Availability Considerations 24
Managing Virtual Machines and Containers 25
Creating Virtual Machines and Containers 25
Supported Guest Operating Systems 27
Choosing OS EZ Template 28
Performing Initial Configuration 28
Configuring Network Settings 29
Setting Passwords for Virtual Machines and Containers 29

Setting Startup Parameters 30
Starting, Stopping, and Querying Status of Virtual Machines and Containers 30
Listing Virtual Machines and Containers 31
Storing Extended Information on Virtual Machines and Containers 32
Copying Virtual Machines and Containers within Server 32
Suspending Virtual Machines and Containers 34
Running Commands in Virtual Machines and Containers 34
Deleting Virtual Machines and Containers 35
Viewing Detailed Information About Virtual Machines and Containers 35
Managing Virtual Machine and Container Backups 37
Backups Overview 37
Using prlctl backup and prlctl restore 38
Using pbackup and prestore 41
Managing Templates 44
Creating Templates 45
Listing Templates 45
Deploying Templates 45
Managing Snapshots 46
Creating Snapshots 46
Listing Snapshots 48
Reverting to Snapshots 49
Deleting Snapshots 49
Migrating Virtual Machines and Containers 49
General Migration Requirements 50
Migrating Virtual Machines and Containers Between Parallels Servers 51
Migrating Containers to Virtual Machines 54
Migrating Physical Computers to Virtual Machines and Containers 56
Migrating Virtual Machines to Containers 60
Migrating Xen Virtual Machines 61
Performing Container-Specific Operations 62
Reinstalling Containers 63
Enabling VPN for Containers 64
Performing Virtual Machine-Specific Operations 65
Pausing Virtual Machines 65

Managing Virtual Machine Devices 65
Making Screenshots 72
Assigning USB Devices to Virtual Machines 72
Configuring IP Address Ranges for Host-Only Networks 74
Converting Third-Party Virtual Machines and Disks 75
Managing Resources 77
What are Resource Control Parameters? 77
Managing CPU Resources 77
Configuring CPU Units 78
Configuring CPU Affinity for Virtual Machines and Containers 78
Configuring CPU Limits for virtual machines and Containers 79
Binding CPUs to NUMA Nodes 81
Enabling CPU Hotplug for Virtual Machines 82
Managing Disk Quotas 83
What are Disk Quotas? 83
Disk Quota Parameters 83
Managing Per-Container Disk Quotas 84
Managing Per-User and Per-Group Disk Quotas 85
Managing Virtual Disks 86
Changing Virtual Machine Disk Type 86
Increasing Disk Capacity 87
Reducing Disk Capacity 88
Compacting Disks 89
Managing Virtual Machine Disk Interfaces 89
Managing Network Accounting and Bandwidth 90
Network Traffic Parameters 90
Configuring Network Classes 91
Viewing Network Traffic Statistics 92
Turning On and Off Network Bandwidth Management 93
Configuring Network Bandwidth Management 94
Managing Disk I/O Parameters 96
Configuring Priority Levels for Virtual Machines and Containers 96
Configuring Disk I/O Bandwidth 97
Configuring the Number of I/O Operations Per Second 98
Viewing Disk I/O Statistics 98

Setting Disk I/O Limits for Container Backups and Migrations 99
Locating Disk I/O Bottlenecks for Containers 100
Managing Memory Parameters for Containers 101
Configuring Main VSwap Parameters 101
Configuring the Memory Allocation Limit 102
Configuring OOM Killer Behavior 103
Tuning VSwap 104
Configuring Legacy Containers 104
Managing Memory Parameters for Virtual Machines 105
Configuring Main Memory Parameters 105
Configuring Additional Memory Parameters 106
Enabling Memory Hotplug for Virtual Machines 109
Managing Container Resources Configuration 110
Splitting Server Into Equal Pieces 111
Scaling Container Configuration 111
Applying New Configuration Sample to Container 112
Managing Virtual Machine Configuration Samples 113
Monitoring Resources 114
Managing Services and Processes 116
What Are Services and Processes 116
Main Operations on Services and Processes 117
Managing Processes and Services 118
Viewing Active Processes and Services 118
Monitoring Processes in Real Time 120
Determining Container Identifiers by Process IDs 121
Managing Parallels Cloud Server Network 122
Managing Network Adapters on the Parallels Server 122
Listing Adapters 123
Creating VLAN Adapters 123
Networking Modes in Parallels Cloud Server 124
Container Network Modes 124
Virtual Machine Network Modes 128
Differences Between Host-Routed and Bridged Network Modes 131
Configuring Virtual Machines and Containers in Host-Routed Mode 131

Configuring Virtual Machines and Containers in Bridged Mode 133
Managing Virtual Networks 133
Managing Adapters in Containers 138
Managing Adapters in Virtual Machines 141
Managing Private Networks 143
Learning Private Networks 143
Setting Up Private Networks 146
Configuring Offline Management 148
Understanding Offline Management 149
Enabling and Disabling Offline Management 152
Enabling and Disabling Offline Services 153
Offline Management Configuration Files 153
Using Open vSwitch Bridges 154
Managing Licenses 156
Installing the License 156
Updating the Current License 157
Transferring the License to Another Server 158
Viewing the Current License 159
Viewing the License 159
License Statuses 161
Keeping Your System Up To Date 162
Updating Parallels Cloud Server 162
Rebootless Updates 162
Updating All Components 163
Updating Kernel 163
Updating EZ Templates 164
Checking for Updates 164
Performing More Actions 164
Updating Software In Virtual Machines 164
Updating Containers 165
Updating EZ Template Packages Inside a Container 165
Updating OS EZ Template Caches 166
Advanced Tasks 168
Configuring Capabilities 168

Available Capabilities for Containers 168
Creating Customized Containers 170
Using OS Template Caches with Preinstalled Application Templates 170
Using Customized OS EZ Templates 172
Using EZ OS Template Sets 173
Using Customized Application Templates 175
Changing System Time from Containers 176
Obtaining Server ID from Inside a Container 177
Restarting Containers 177
Enabling VNC Access to Virtual Machines and Containers 178
Enabling VNC Access to Virtual Machines 178
Enabling VNC Access to Containers 178
Connecting with a VNC Client 179
Setting Immutable and Append Flags for Container Files and Directories 179
Managing iptables Modules 179
Using iptables Modules on Servers 179
Using iptables Modules in Containers 180
Creating Configuration Files for New Linux Distributions 181
Aligning Disks and Partitions in Virtual Machines 182
Running Parallels Cloud Server 6.0 in Virtual Machines 187
Installing Optional Parallels Cloud Server Packages 189
Working with Shared Folders in Linux Virtual Machines 189
Sharing with Ownership of a Virtual Machine User 189
Mounting as a Specific User 191
Sharing with Ownership of a Server User 191
Legacy Features 193
Using Virtuozzo File System 193
Creating VZFS-based Containers 193
Converting VZFS Containers to New Layout 194
Listing Legacy Container Backups 194
Troubleshooting 195
General Considerations 195
Kernel Troubleshooting 197
Using ALT+SYSRQ Keyboard Sequences 197

Saving Kernel Faults (OOPS) 197
Finding a Kernel Function That Caused the D Process State 199
Problems with Container Management 199
Failure to Start a Container 199
Failure to Access a Container from Network 200
Failure to Log In to a Container 200
Getting Technical Support 201
Preparing and Sending Questions to Technical Support 201
Submitting a Problem Report to Technical Support 202
Establishing Secure Channel to Parallels Support 202
Glossary 204
Index 206

CHAPTER 1

Introduction

Parallels Cloud Server 6.0 is a virtualization solution that allows you to run multiple virtual machines and Containers on a single physical server.

This chapter provides general information about Parallels Cloud Server and this guide. You will learn

- goals and target audience of the guide (p. 10)
- guide organization (p. 10)
- documentation conventions used in the guide (p. 11)
- resources to consult to get more information on Parallels Cloud Server (p. 12)
- ways to submit feedback to the Parallels documentation team (p. 12)

In This Chapter

About This Guide 10
Getting Help 12
Feedback 12

About This Guide

The *Parallels Cloud Server 6.0 User's Guide* provides comprehensive information on Parallels Cloud Server 6.0, high-end virtualization software for bare metal servers. The guide covers the necessary theoretical concepts as well as practical aspects of working with Parallels Cloud Server.

Note: The guide does not explain how to install and configure your Parallels Cloud Server 6.0 system. For detailed information on these operations, see the *Parallels Cloud Server 6.0 Installation Guide*.

The primary audience for this guide is administrators responsible for managing systems that run Parallels Cloud Server 6.0. The guide assumes that you have a working knowledge of Linux operating systems (command line, system administration, and so on).

Organization of This Guide

This guide is organized in the following way:

Chapter 1, Introduction (p. 10), gives an overview of the Parallels Cloud Server product and this guide.

Chapter 2, Learning Parallels Cloud Server 6.0 Basics (p. 14), explains the general principles of Parallels Cloud Server operation.

Chapter 3, Managing Virtual Machines and Containers (p. 25), covers those operations that you can perform on a virtual machine and Container: creating and deleting virtual machines and Containers, starting and stopping them, backing up and restoring, etc. You will also learn how to perform different kinds of migration: migrate virtual machines and Containers between Parallels servers, migrate a physical server to a virtual machine and Container, and migrate a Container to a virtual machine.

Chapter 4, Managing Resources (p. 77), focuses on configuring and monitoring the resource control parameters for virtual machines and Containers. These parameters comprise disk quotas, network accounting and shaping, CPU and system resources.

Chapter 5, Managing Services and Processes (p. 116), familiarizes you with the operations you can perform on processes and services in Parallels Cloud Server.

Chapter 6, Managing Parallels Cloud Server Network (p. 122), familiarizes you with the Parallels Cloud Server network structure and explains how to manage networks in Parallels Cloud Server systems.

Chapter 7, Managing Licenses (p. 156), provides detailed information on managing licenses in Parallels Cloud Server.

Chapter 8, Keeping Your System Up To Date (p. 162), informs you of the ways to keep all the software components of a Parallels server up to date.

Chapter 9, Advanced Tasks (p. 168), enumerates those tasks that are intended for advanced system administrators who would like to obtain deeper knowledge about Parallels Cloud Server capabilities.

Chapter 10, Troubleshooting (p. 195), suggests ways to resolve common inconveniences should they occur during your work with Parallels Cloud Server.

Documentation Conventions

Before you start using this guide, it is important to understand the documentation conventions used in it.

The table below presents the existing formatting conventions.

Formatting convention	Type of Information	Example
Special Bold	Items you must select, such as menu options, command buttons, or items in a list.	Go to the **Resources** tab.
	Titles of chapters, sections, and subsections.	Read the **Basic Administration** chapter.

Italics	Used to emphasize the importance of a point, to introduce a term or to designate a command-line placeholder, which is to be replaced with a real name or value.	These are the so-called *OS templates*. To remove a Container, type `prlctl delete` *`ctid`*.
`Monospace`	The names of commands, files, and directories.	Use `prlctl start` to start a Container.
`Preformatted`	On-screen computer output in your command-line sessions; source code in XML, C++, or other programming languages.	`Saved parameters for Container 101`
`Monospace Bold`	What you type, as contrasted with on-screen computer output.	**`C:\vzlist -a`**
Key+Key	Key combinations for which the user must press and hold down one key and then press another.	Ctrl+P, Alt+F4

Besides the formatting conventions, you should also know about the document organization convention applied to Parallels documents: chapters in all guides are divided into sections, which, in their turn, are subdivided into subsections. For example, **About This Guide** is a section, and **Documentation Conventions** is a subsection.

Getting Help

In addition to this guide, there are a number of other resources available for Parallels Cloud Server which can help you use the product more effectively. These resources include:

- *Parallels Cloud Server 6.0 Installation Guide*. This guide provides detailed information on installing Parallels Cloud Server on your server, including the pre-requisites and the stages you shall pass.
- *Getting Started With Parallels Cloud Server 6.0***.** This guide provides basic information on how to install Parallels Cloud Server on your server, create new Containers and virtual machines, and perform main operations on them. As distinct from the *Parallels Cloud Server 6.0 Installation Guide*, it does not contain detailed description of all the operations needed to install and set Parallels Cloud Server to work (e.g., installing Parallels Cloud Server in text mode).
- *Parallels Cloud Server 6.0 Templates Management Guide*. This guide is meant to provide complete information on Parallels templates - an exclusive Parallels technology allowing you to efficiently deploy standard Linux applications inside your Containers and to greatly save the physical server resources (physical memory, disk space, etc.).
- *Parallels Cloud Server 6.0 Command Line Reference Guide*. This guide is a complete reference on all Parallels Cloud Server configuration files and command line utilities.

Feedback

If you spot a typo in this guide, or if you have an opinion about how to make this guide more helpful, you can share your comments and suggestions with us by completing the Documentation Feedback form on our website (http://www.parallels.com/en/support/usersdoc/).

CHAPTER 2

Learning Parallels Cloud Server 6.0 Basics

This chapter provides a brief description of Parallels Cloud Server 6.0, Parallels virtual machines and Containers, their specifications and underlying technologies.

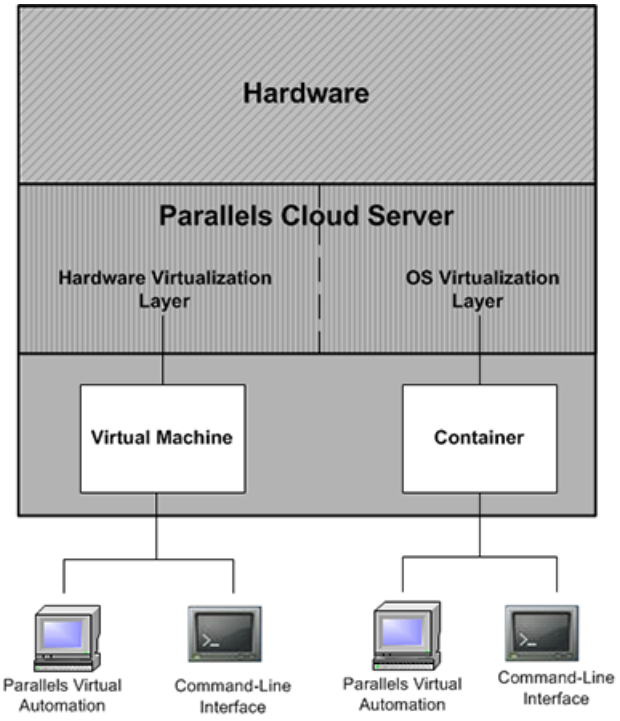
In This Chapter

Parallels Cloud Server 6.0 Overview 14
OS Virtualization Layer 16
Hardware Virtualization Layer 19
Parallels Cloud Server Configuration 22
Resource Management 23
Understanding Licensing 23
Physical Server Availability Considerations 24

Parallels Cloud Server 6.0 Overview

Parallels Cloud Server 6.0 allows you to simultaneously run multiple Parallels virtual machines and Containers on a single physical server. Using this software, you can efficiently use your server's hardware resources by sharing them among virtual machines and Containers.

Graphically, a server with the Parallels Cloud Server software installed can be represented as follows:



At the base resides server hardware. Next is the Parallels Cloud Server software which is installed directly on the server hardware and does not need any operating system for its functioning. Parallels Cloud Server includes two virtualization layers:

- **Hardware virtualization layer**. This layer provides the necessary environment for creating and managing Parallels virtual machines.
- **OS virtualization layer**. This layer provides the necessary environment for creating and managing Parallels Containers.

For more information on both layers, see **OS Virtualization Layer** (p. 16) and **Hardware Virtualization Layer** (p. 19).

Effectively uniting both virtualization technologies, Parallels Cloud Server provides the best value for cost conscious organizations enabling them to:

- standardize server hardware platforms
- effectively consolidate server resources
- consolidate and support legacy OSs and applications
- streamline server and application deployment, maintenance, and management
- simplify software testing and development
- optimize server and application availability

You can use the following tools to manage virtual machines and Containers:

- **Command line interface (CLI)**. This tool comprises a set of Parallels command line utilities and

can be used to manage virtual machines and Containers both locally and remotely.

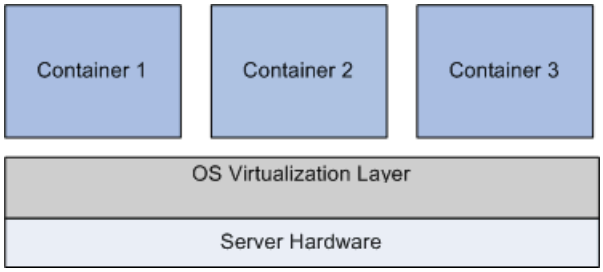
- **Parallels Virtual Automation**. Parallels Virtual Automation is a tool for managing physical servers, Containers, and virtual machines with the help of a standard Web browser on any platform.

OS Virtualization Layer

This section provides detailed information on the OS virtualization layer, one of the two components of Parallels Cloud Server, responsible for providing support for Parallels Containers.

Basics of OS Virtualization

The OS virtualization allows you to virtualize physical servers on the operating system (kernel) layer. The diagram below shows the basic architecture of OS virtualization.



The OS virtualization layer ensures isolation and security of resources between different Containers. The virtualization layer makes each Container appear as a standalone server. Finally, the Container itself houses its own applications and workload. OS virtualization is streamlined for the best performance, management, and efficiency. Its main advantages are the following:

- Containers perform at levels consistent with native servers. Containers have no virtualized hardware and use native hardware and software drivers making its performance unbeatable.
- Each Container can seamlessly scale up to the resources of an entire physical server.
- OS virtualization technology provides the highest density available from a virtualization solution. You can create and run up to 100s of Containers on a standard production physical server.
- Containers use a single OS, making it extremely simple to maintain and update across Containers. Applications may also be deployed as a single instance.

Parallels Containers

From the point of view of applications and Container users, each Container is an independent system. This independence is provided by the Parallels Cloud Server OS virtualization layer. Note that only a negligible part of the CPU resources is spent on virtualization (around 1-2%). The main features of the virtualization layer implemented in Parallels Cloud Server are the following:

- A Container looks like a normal Linux system. It has standard startup scripts; software from vendors can run inside Containers without any modifications or adjustment.
- A user can change any configuration file and install additional software inside Containers.
- Containers are fully isolated from each other (file system, processes, sysctl variables) and Parallels virtual machines.
- Containers share dynamic libraries, which greatly saves memory.
- Processes belonging to a Container are scheduled for execution on all available CPUs. Consequently, Containers are not bound to only one CPU and can use all available CPU power.

The two key parts of any Container are the contents and configuration. By default, all Container files are stored in the `/vz/private/<CT_ID>` directory on the Parallels server, also called *private area*.

Key Container directories and files:

File Name	Description
`/vz/private/<CT_ID>`	Container private area.
`/vz/private/<CT_ID>/root.hdd/root.hdd`	Virtual hard disk with Container contents.
`/vz/root/<CT_ID>`	Container mount point.
`ve.conf`	Container configuration file: • Is symlinked to `/etc/vz/conf/<CT_ID>.conf`. • Defines Container parameters, such as allocated resource limits, IP address and hostname, and so on. • Overrides matching parameters in the global configuration file.

All Container files are stored in a single image (`/vz/private/<CT_ID>/root.hdd/root.hdd`), similar to a virtual machine's hard disk. Such standalone nature:

- Enables easier migrations and backups due to a faster sequential I/O access to Container images than to separate Container files.
- Removes the need for OS and application templates once a Container is created.
- Allows the use of native Linux disk quotas that are journaled and does not require quota recalculation after disasters like server crashes.

Note: Using Containers that store all files in an image file (also known as Containers with the *Container-in-an-image-file* layout) is supported only for `/vz` partitions formatted as ext4.

Memory and IOPS Deduplication

Parallels Cloud Server supports memory and IOPS deduplication that helps eliminate memory and I/O bottlenecks in the new image-based Containers and increases the number of running Containers per Parallels server.

Identical files in multiple Containers are efficiently cached and the resulting cache can be stored on an SSD drive to further improve overall system performance and reduce the starting times of multiple similar Containers.

Note: Memory and IOPS deduplication can be managed with the `pfcache` utility described in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Templates

A template (or a package set) in Parallels Cloud Server is a set of original application files repackaged for use by Parallels Cloud Server. Usually, it is just a set of RPM packages for Red Hat like systems. Parallels Cloud Server provides tools for creating templates, installing, upgrading, adding them to and removing them from a Container.

Using templates lets you:

- Share RAM among similar applications running in different Containers to save hundreds of megabytes of memory.
- Deploy applications simultaneously in many Containers.
- Use different versions of an application in different Containers (for example, perform upgrades only in certain Containers).

There are two types of templates: OS and application:

- An OS template is an operating system and the standard set of applications to be found right after the installation. Parallels Cloud Server uses OS templates to create new Containers with a preinstalled operating system.
- An application template is a set of repackaged software packages optionally accompanied with configuration scripts. Application templates are used to add extra software to existing Containers.

For example, you can create a Container on the basis of the `redhat` OS template and add the MySQL application to it with the help of the `mysql` template.

Note: For detailed information on Parallels templates, see the *Parallels Cloud Server 6.0 Templates Management Guide*.

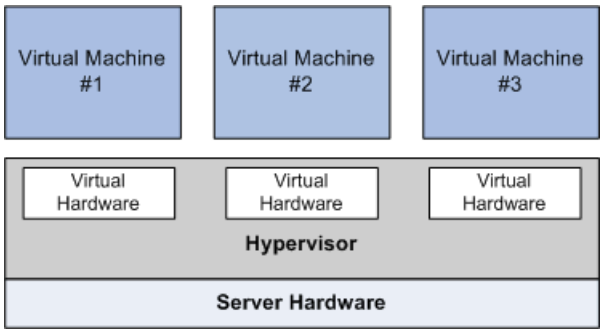
Hardware Virtualization Layer

This section familiarizes you with the second component of Parallels Cloud Server - the hardware virtualization layer. This layer provides the necessary environment for creating and managing Parallels virtual machines.

Basics of Hardware Virtualization

Parallels Cloud Server is based on the concept of *hardware virtualization*. Hardware virtualization has a base layer—a hypervisor. This layer is loaded directly on the bare server and acts as an intermediary between the server hardware and virtual machines. To allocate hardware and resources to virtual machines, Parallels Cloud Server virtualizes all hardware on the server. Once virtualized, hardware and resources can be easily assigned to virtual machines. Based on the virtual hardware, a virtual machine runs its own complete copies of an operating system and applications.

The following diagram shows the basic architecture of hardware virtualization.



Like OS virtualization, hardware virtualization also provides many benefits the main of which are listed below:

- Create multiple virtual machines with different operating systems on a single physical computer.
- Run several guest operating systems and their applications simultaneously on a single physical computer without rebooting.
- Consolidate and virtualize the computing environment, reduce hardware costs, lower operating expenses, and increase productivity.
- Use open APIs and SDK to extend management integration with in-house and third-party applications.

Parallels Virtual Machines

From the point of view of applications and virtual machine users, each virtual machine is an independent system with an independent set of virtual hardware. This independence is provided by

the Parallels Cloud Server hardware virtualization layer. The main features of the virtualization layer are the following:

- A virtual machine looks like a normal computer. It has its own virtual hardware, and software applications can run in virtual machines without any modifications or adjustment.
- A user can easily change the virtual machine configuration (e.g., add a new virtual disk or increase memory).
- Virtual machines are fully isolated from each other (file system, processes, `sysctl` variables) and Parallels Cloud Server.
- Install any of the supported operating systems in the virtual machine. The guest operating system and its applications are isolated inside a virtual machine and share physical hardware resources with other virtual machines.

Intel and AMD Virtualization Technology Support

Parallels Cloud Server provides support for Intel and AMD virtualization technologies comprising a set of processor enhancements and improving the work of virtualization solutions. Utilizing these technologies, Parallels Cloud Server can offload some workload to the system hardware, which results in the "near native" performance of guest operating systems.

Virtual Machine Hardware

A Parallels virtual machine works like a stand-alone computer with the following hardware:

CPU	Up to 32 Intel/AMD CPUs
Motherboard	Intel i965-based motherboard
RAM	Up to 128 GB of RAM
Video Adapter	VGA/SVGA video adapter with VBE 3.0
Video RAM	Up to 256 MB of video memory
Floppy Disk Drive	1.44 MB floppy disk drive mapped to an image file or a physical floppy drive
SATA Devices • **Hard Disk** • **CD/DVD-ROM Drive**	Up to 6 SATA devices Hard disk drive mapped to an image file (up to 5 TB each) CD/DVD-ROM drive mapped to a physical drive or an image file
SCSI Devices • **Hard Disk** • **Generic SCSI Device**	Up to 15 SCSI devices Hard disk drive mapped to an image file (up to 5 TB each) Generic SCSI device
Network Interfaces	Up to 16 Intel e1000 virtual network adapters.
Serial (COM) Ports	Up to 4 serial (COM) ports mapped to a socket, a real port, or an output file
Parallel (LPT) Ports	Up to 3 parallel (LPT) ports mapped to a printer, a real port, or an output file
Sound Card	AC'97-compatible sound card, sound recording support
Keyboard	Generic USB, PS/2 keyboard

Mouse	Generic USB, PS/2 wheel mouse

Virtual Machine Files

A virtual machine has at least two files: a configuration file (PVS file) and a hard disk image file (HDD file). It can also have additional files: a file for each additional virtual hard disk and output files for virtual ports. By default, the virtual machines files are stored in the `/var/parallels` directory on the Parallels server.

The list of files related to a virtual machine is given in the table below:

File Name	Description
`.pvm`	A bundle that contains the virtual machine files.
`.pvs`	A virtual machine configuration file. It defines the hardware and resources configuration of the virtual machine. The configuration file is automatically generated during the virtual machine creation.
`.sav`	A dump file created when you suspend the virtual machine. This file contains the state of the virtual machine and its applications at the moment the suspend was invoked.
`.mem`	A file containing the memory dump for the suspended virtual machine. For a running virtual machine, it is a temporary virtual memory file.
`.hdd`	A file representing a virtual hard disk. When you create a virtual machine, you can create it with a new virtual hard disk or use an existing one. A virtual machine can have several hard disks.
`.iso`	An image file of a CD or DVD disc. Virtual machines treat ISO images as real CD/DVD discs.
`.txt`	Output files for serial and parallel ports. The output `.txt` files are generated when a serial or parallel port connected to an output file is added to the virtual machine configuration.

Support of Virtual and Real Media

This section lists the types of disks that can be used by Parallels virtual machines and provides the information about basic operations you can perform on these disks.

Supported Types of Hard Disks

Parallels virtual machines can use only virtual hard disks image files as their hard disks.

Virtual Hard Disks

The capacity of a virtual hard disk can be set from 100 MB to 5 TB.

Virtual hard disks can be of either *plain* or *expanding* format. When you create a virtual machine in **Express Windows** or **Typical** mode (in the New Virtual Machine wizard), the disk is created in the *expanding* format.

Type	Description
plain	A plain virtual hard disk image file has a fixed size. The size is determined when the disk is created. Plain disks can be created with the help of New Virtual Machine wizard (the **Custom** mode.)
expanding	An expanding virtual hard disk image file is small initially. Its size grows as you add applications and data to the virtual hard disk in the guest OS.

Split disks

A virtual disk of either format can be a single-piece disk or a split disk. A split disk is cut into 2 GB pieces and is stored as a single `.hdd` file.

CD/DVD Discs and Their Images

Parallels Cloud Server can access real CD/DVD discs and images of CD/DVD discs.

Parallels Cloud Server has no limitations on using multi-session CD/DVD discs. A virtual machine can play back audio CDs without any limitations on copy-protected discs.

If your server has a recordable optical drive, you can use it to burn CD or DVD discs in a virtual machine.

Parallels Cloud Server supports CD/DVD disc images in ISO, CUE, and CCD formats.

Floppy Disks and Floppy Disk Images

Parallels Cloud Server can use two types of floppy disks:

- Real diskettes inserted into a real floppy disk drive that is connected to the virtual machine.
- Floppy disk image files having the `.fdd` extension and connected to the virtual machine.

Parallels Cloud Server treats floppy disk images like real diskettes. Parallels Cloud Server supports floppy disk image files that have the `.fdd` extension and are 1.44 MB in size.

With Parallels Cloud Server, you can also create an image of a blank floppy using the Floppy Disk pane of the **Virtual Machine Configuration** dialog.

Note: Parallels Cloud Server cannot create images of real diskettes.

Parallels Cloud Server Configuration

Parallels Cloud Server allows you to configure settings for the physical server in general and for each Container in particular. Among these settings are disk and user quotas, network parameters, default file locations, sample configuration files, and other.

Parallels Cloud Server stores all OS virtualization-related configuration information in the global configuration file `/etc/vz/vz.conf`. It defines Container parameters like the default OS templates, disk quotas, logging, and so on. For a list of parameters constituting the global configuration file, refer to the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

The configuration file is read when the Parallels Cloud Server software and/or Containers are started. However, many settings can also be changed on the fly by means of Parallels Cloud Server standard utilities like `prlctl`, with or without modifying the corresponding configuration file to keep the changes for the future.

Resource Management

Parallels Cloud Server resource management controls the amount of resources available to virtual machines and Containers. The controlled resources include such parameters as CPU power, disk space, a set of memory-related parameters. Resource management allows you to:

- effectively share available physical server resources among virtual machines and Containers
- guarantee Quality-of-Service in accordance with a service level agreement (SLA)
- provide performance and resource isolation and protect from denial-of-service attacks
- simultaneously assign and control resources for a number of virtual machines and Containers
- collect usage information for system health monitoring

Resource management is much more important for Parallels Cloud Server than for a standalone server since server resource utilization in such a system is considerably higher than that in a typical system.

Understanding Licensing

To start using the Parallels Cloud Server software, you need a special license - *Parallels Cloud Server license*. You must install this license on your server after or when installing Parallels Cloud Server on it. Every physical server hosting virtual machines and Containers must have its own license. Licenses are issued by Parallels and define a number of parameters in respect of your physical server. The main licensed parameters are listed below:

- The number of CPUs which can be installed on the physical server. Keep in mind that each of the Dual Core and Hyperthreading processors is regarded as one CPU.
- The license expiration date. Any license can be time-limited or permanent.

  Parallels Cloud Server licenses have a start date, and if they are time-limited, can also have an

expiration date specified in them. You must set up your system clock correctly; otherwise, the license validation may fail.

- The number of virtual machines and Containers that can simultaneously run on he physical server.
- The platform and architecture with which the Parallels Cloud Server software is compatible.

Physical Server Availability Considerations

The availability of a physical server running Parallels Cloud Server is more critical than the availability of a typical PC server. Since it runs multiple virtual machines and Containers providing a number of critical services, physical server outage might be very costly. It can be as disastrous as the simultaneous outage of a number of servers running critical services.

To increase physical server availability, we suggest that you follow the recommendations below:

- Use a RAID storage for critical virtual machines and Containers. Do prefer hardware RAIDs, but software mirroring RAIDs might suit too as a last resort.
- Do not run any software on the server itself. Create special virtual machines and Containers where you can host necessary services such as BIND, FTPD, HTTPD, and so on. On the server, you need only the SSH daemon. Preferably, it should accept connections from a pre-defined set of IP addresses only.
- Do not create users on the server itself. You can create as many users as you need in any virtual machine and Container. Remember: compromising the server means compromising all virtual machines and Containers as well.

CHAPTER 3

Managing Virtual Machines and Containers

This chapter describes how to perform day-to-day operations on virtual machines and Containers.

Note: We assume that you have successfully installed, configured, and deployed your Parallels Cloud Server system. If you have not, refer to the *Parallels Cloud Server 6.0 Installation Guide* for detailed information on these operations.

In This Chapter

Creating Virtual Machines and Containers 25
Performing Initial Configuration 28
Starting, Stopping, and Querying Status of Virtual Machines and Containers..... 30
Listing Virtual Machines and Containers 31
Storing Extended Information on Virtual Machines and Containers..... 32
Copying Virtual Machines and Containers within Server..... 32
Suspending Virtual Machines and Containers..... 34
Running Commands in Virtual Machines and Containers..... 34
Deleting Virtual Machines and Containers..... 35
Viewing Detailed Information About Virtual Machines and Containers..... 35
Managing Virtual Machine and Container Backups..... 37
Managing Templates..... 44
Managing Snapshots..... 46
Migrating Virtual Machines and Containers..... 49
Performing Container-Specific Operations..... 62
Performing Virtual Machine-Specific Operations 65

Creating Virtual Machines and Containers

This section explains how to create new Parallels virtual machines and Containers. The options you should pass to this command differ depending on whether you want to create a virtual machine or Container.

Creating a Container

To create a Container, use the `prlctl create` command. The following parameters are required:

Argument	Description
Container name	Name to assign to the Container. A name may contain digits and letters. When using digits only, start with numbers greater than 100 (101, 102, etc.).
OS template name	The name of the OS template to base your Container on. Parallels Cloud Server is shipped with a number of ready-to-use OS templates. To find out the names of the available templates, use the `vzpkg list -O` command. For the list of operating systems you can run in your Virtual Machines and Containers, see **Supported Guest Operating Systems** (p. 27).
Type of virtual server	Tells the `prlctl create` command to create a Container.

Thus, for example, you can create a new Container by executing the following command:

```
# prlctl create 101 --ostemplate centos-6-x86 --vmtype ct
```

In this case Parallels Cloud Server creates a new Container with the name of `101` and the CentOS operating system installed inside. All Container contents will be stored in its private area. To find out where the Container private area is located, use the `prlctl list` command as follows:

```
# prlctl list 101 -i | grep "Home"
Home: /vz/private/101
```

Notes:

1. For more information on options you can pass to `prlctl create` when creating Containers, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

2. For information on creating Containers with preinstalled applications, see **Using OS Template Caches with Preinstalled Applications** (p. 170).

Creating a Virtual Machine

The process of creating a new virtual machine includes the following steps:

1. Creating a virtual machine configuration.
2. Installing an operating system in the virtual machine.
3. Installing Parallels Tools in the virtual machine, a set of special utilities that facilitate your work with virtual machines.

The example below shows you how to make a new virtual machine configuration using the `prlctl create` command:

```
# prlctl create MyVM --distribution win-2008 --vmtype vm
```

This creates a virtual machine with the name of `MyVM`, adjusts its configuration for installing the Windows Server 2008 operating system in it, and places all virtual machine-related files in the `/var/parallels` directory.

Once the virtual machine configuration is ready, you can install Windows Server 2008 and Parallels Tools in it. To do this, you can

- Use the Parallels Virtual Automation application. For details on using this application, consult the *Parallels Virtual Automation Administrator's Guide*.
- Enable VNC support in the virtual machine configuration and install the operating system and Parallels Tools using your favorite VNC client. For information, on configuring VNC support in virtual machines, see **Enabling VNC Access to Virtual Machines and Containers** (p. 178).

Note: For more information on options you can pass to `prlctl create` when creating virtual machines, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Supported Guest Operating Systems

Listed below are the operating systems that have been tested in virtual machines and Containers and are officially supported in Parallels Cloud Server 6.0.

Virtual Machines

Windows

Windows 8 (x64)

Windows 7 with Service Pack 1 (x86, x64)

Windows Server 2012

Windows Server 2008 R2 with Service Pack 1 (x64)

Windows Server 2003 R2 with Service Pack 2 (x86, x64)

Windows XP Professional with Service Pack 2 (x64) and Service Pack 3 (x86)

Linux

Red Hat Enterprise Linux 6.x (x86, x64)

Red Hat Enterprise Linux 5.x (x86, x64)

Fedora 17 (x86, x64)

Fedora 16 (x86, x64)

CentOS 6.x (x86, x64)

CentOS 5.x (x86, x64)

SUSE Linux Enterprise Server 11 with Service Pack 2 (x86, x64)

SUSE Linux Enterprise Server 10 (x86, x64)

openSUSE 12.x (x86, x64)

Debian GNU/Linux 6.0 (x86, x64)

Ubuntu 12.10 (x86, x64)

Ubuntu 11.10 (x86, x64)

Ubuntu Server 12.04 (x86, x64)

Ubuntu Server 10.04 (x86, x64)

FreeBSD

FreeBSD 9 (x86, x64)

Containers

Red Hat Enterprise Linux 6.x (x86, x64)

Red Hat Enterprise Linux 5.x (x86, x64)

Fedora 17 (x86, x64)

Fedora 16 (x86, x64)

CentOS 6.x (x86, x64)

CentOS 5.x (x86, x64)

SUSE Linux Enterprise Server 11 with Service Pack 2 (x86, x64)

openSUSE 12.1 (x86, x64)

openSUSE 11.4 (x86, x64)

Debian GNU/Linux 6.0 (x86, x64)

Ubuntu 12.10 (x86, x64)

Ubuntu 12.04 (x86, x64)

Ubuntu 11.10 (x86, x64)

Ubuntu 10.04 (x86, x64)

Choosing OS EZ Template

Before starting to create a Container, you shall decide on which OS EZ template your Container will be based. There might be several OS EZ templates installed on the server and prepared for the Container creation; use the `vzpkg list` command to find out what OS EZ templates are available on your system:

```
# vzpkg list -O
centos-6-x86
centos-6-x86_64                    2012-05-10 13:16:43
```

Using the `-O` option with the `vzpkg list` command, you can list only the OS EZ templates installed on the server. The time next to an OS EZ template indicates when the template was cached.

You can also use the `--with-summary` option to display brief information on the installed OS EZ templates:

```
# vzpkg list -O --with-summary
centos-6-x86                       :CentOS 6 EZ OS template
centos-6-x86_64                    :CentOS 6 (for AMD64/Intel EM64T) EZ OS Template
```

For detailed information on the `vzpkg list` command, consult the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Performing Initial Configuration

Before starting your newly created virtual machine and Container, you first need to configure it. This section describes the main configuration steps for virtual machines and Containers.

Configuring Network Settings

To make virtual machines and Containers accessible from the network, you need to assign valid IP addresses to them and configure DNS servers. The session below illustrates setting these parameters for the `MyVM` virtual machine and Container 101:

- Assigning IPv4 and IPv6 addresses:

```
# prlctl set MyVM --device-set net0 --ipadd 10.0.186.100/24
# prlctl set MyVM --device-set net0 --ipadd 1fe80::20c:29ff:fe01:fb07
# prlctl set 101 --ipadd 10.0.186.101/24
# prlctl set 101 --ipadd fe80::20c:29ff:fe01:fb08
```

  `net0` in the commands above denotes the network card in the `VM` virtual machine to assign the IP address to. You can view all network cards of a virtual machine using the `prlctl list` *`VM_name`* `-i` command.

- Setting DNS server addresses:

```
# prlctl set MyVM --nameserver 192.168.1.165
# prlctl set 101 --nameserver 192.168.1.165
```

Notes:

1. You can configure the network settings only for virtual machines that have Parallels Tools installed.

2. To assign network masks to Containers operating in the `venet0` network mode, you must set the `USE_VENET_MASK` parameter in the `/etc/vz/vz.conf` configuration file to `yes`.

Setting Passwords for Virtual Machines and Containers

In Parallels Cloud Server, you can use the `--userpasswd` option of the `prlctl set` command to create new accounts in your virtual machines and Containers directly from the Parallels server. The created account can then be used to log in to the virtual machine and Container. The easiest way of doing it is to run this command:

```
# prlctl set MyVM --userpasswd user1:2wsx123qwe
```

This command creates the `user1` account in the `MyVM` virtual machine and sets the `2wsx123qwe` password for it. Now you can log in to the `MyVM` virtual machine as `user1` and administer it in the same way you would administer a standalone server: install additional software, add users, set up services, and so on.

The `prlctl set` command can also be used to change passwords for existing accounts in your virtual machines and Containers. For example, to change the password for `user1` in the MyVM virtual machine to `0pi65jh9`, run this command:

```
# prlctl set MyVM --userpasswd user1:0pi65jh9
```

When setting passwords for virtual machines and Containers, keep in mind the following:

- You can use manage user accounts only inside virtual machines that have Parallels Tools installed.
- You should use passwords that meet the minimum length and complexity requirements of the respective operating system. For example, for Windows Server 2008, a password must be more than six characters in length and contain characters from three of the following categories: uppercase characters, lowercase characters, digits, and non-alphabetic characters.
- You should not create accounts with empty passwords for virtual machines and Containers running Linux operating systems.

Setting Startup Parameters

The `prlctl set` command allows you to define the `onboot` startup parameter for virtual machines and Containers. Setting this parameter to `yes` makes your virtual machine and Container automatically boot at the physical server startup. For example, to enable Container 101 and the `MyVM` virtual machine to automatically start on your server boot, you can execute the following commands:

- For Container 101:

```
# prlctl set 101 --onboot yes
```

- For the `MyVM` virtual machine:

```
# prlctl set MyVM --onboot yes
```

Notice that the `onboot` parameter will have effect only on the next server startup.

Starting, Stopping, and Querying Status of Virtual Machines and Containers

After a virtual machine and Container has been created, it can be managed like a usual computer.

Starting Virtual Machines and Containers

You can use the `prlctl start` command to start your virtual machines and Containers:

- To start Container 101:

```
# prlctl start 101
```

- To start a virtual machine with the name of `MyVM`:

```
# prlctl start MyVM
```

Stopping Virtual Machines and Containers

The `prlctl stop` command is used to stop your virtual machines and Containers:

- To stop Container 101:

```
# prlctl stop 101
```

- To stop a virtual machine with the name of `MyVM`:

```
# prlctl stop MyVM
```

Checking Status of Virtual Machines and Containers

To check the status of a Container or a virtual machine, use the `prlctl status` command:

- To get to know the status of Container 101:

```
# prlctl status 101
VEID 101 exists mounted running
```

- To get to know the status of the `MyVM` virtual machine:

```
# prlctl status MyVM
Vm MyVM exists stopped
```

Restarting Virtual Machines and Containers

To restart a virtual machine and Container. To do this, use the `prlctl restart` command:

- To restart a Container:

```
# prlctl restart 101
```

- To restart a virtual machine:

```
# prlctl restart MyVM
```

Listing Virtual Machines and Containers

To get an overview of the virtual machines and Containers existing on the physical server and to get additional information about them — their IP addresses, hostnames, current resource consumption, and so on — use the `prlctl list` command. In the most general case, you can get a list of all virtual machines and Containers by issuing the following command:

```
# prlctl list -a
UUID                                    STATUS     IP_ADDR         T   NAME
{600adc12-0e39-41b3-bf05-c59b7d26dd73}  running    10.10.1.101     CT  101
{b2de86d9-6539-4ccc-9120-928b33ed31b9}  stopped    10.10.100.1     VM  MyVM
```

The `-a` option tells the `prlctl list` command to output both running and stopped virtual machines and Containers. By default, only running virtual machines and Containers are shown. The default columns inform you of the Container IDs and virtual machine names, the virtual machine and Container status, type, IP addresses, and unique identifiers. This output can be customized as desired by using `prlctl list` command line options. For example:

```
# prlctl list -a -o name,ctid
NAME                                    ID
101                                     101
MyVm   {b8cb6d99-1af1-453d-a302-2fddd8f86769}
```

This command displays only the names of existing virtual machines and Containers and the IP addresses assigned to them. The full list of the `prlctl list` command-line options for virtual machines and Containers is available in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Storing Extended Information on Virtual Machines and Containers

Sometimes, it may be difficult to remember the information on certain virtual machines and Containers. The probability of this increases together with the number of virtual machines and Containers and with the time elapsed since their creation. Parallels Cloud Server allows you to set the description of any virtual machine and Container on the physical server and view it later on, if required. The description can be any text containing any virtual machine and Container-related information. For example, you can include the following in the virtual machine and Container description:

- the owner of the virtual machine and Container
- the purpose of the virtual machine and Container
- the summary description of the virtual machine and Container

Let us assume that you are asked to create a virtual machine for a Mr. Johnson who is going to use it for hosting the MySQL server. So, you create the `MyVM` virtual machine and, after that, execute the following command on the physical server:

```
# prlctl set MyVM --description "MyVM
> owner - Mr. Johnson
> purpose - hosting the MySQL server" -
The VM has been successfully configured.
```

This command saves the following information related to the virtual machine: its name, owner, and the purpose of its creation. At any time, you can display this information by issuing the following command:

```
# prlctl list -o description MyVM
MyVM
owner - Mr. Johnson
purpose - hosting the MySQL server
```

When working with virtual machine and Container descriptions, keep in mind the following:

- You can use any symbols you like in the virtual machine and Container description (new lines, dashes, underscores, spaces, etc.).
- If the virtual machine and Container description contains one or more spaces or line breaks (as in the example above), it must be put in single or double quotes.
- As distinct from a virtual machine and Container name and ID, a description cannot be used for performing virtual machine and Container-related operations (e.g., for starting or stopping a virtual machine and Container) and is meant for reference purposes only.

Copying Virtual Machines and Containers within Server

You can create a complete copy of a particular virtual machine and Container (in respect of all the virtual machine and Container data and resources parameters), or a *clone*. This saves your time because you do not have to think of setting up the virtual machine and Container configuration parameters and the like. Moreover, you can create a number of clones at a sitting.

In Parallels Cloud Server, you can use the `prlctl clone` command to copy a virtual machine and Container within the given physical server. To clone a Container or a virtual machine, it must be stopped. For example, you can create a clone of Container 101 and the `MyVM` virtual machine by running these commands:

```
# prlctl clone 101 --name 111
# prlctl clone MyVM --name ClonedVM
```

The `--name` option specifies which ID or name should be assigned to the new clones.

Checking the Cloned Virtual Machine and Container

To check that your virtual machine and Container has been successfully moved, run this command:

```
# prlctl list -a
UUID            STATUS       IP_ADDR         T  NAME
{62951c2a-...}  stopped      10.0.10.101     CT 101
{49b66605-...}  stopped      10.0.10.101     CT 111
{7f4904ad-...}  stopped      10.0.10.115     VM MyVM
{2afb2aa2-...}  stopped      10.30.128.134   VM ClonedVM
```

As you can see from the example above, the clones of Container 101 (Container 111) and the `MyVM` virtual machine (`ClonedVM`) have been successfully created. However, before starting to use the clones, you should assign different IP addresses to them which are currently identical to those of Container 101 and `MyVM`. Refer to **Performing Initial Configuration** (p. 28) to learn how you can do it.

Configuring Default Directories

When cloning a virtual machine and Container, you can also override the following default directories:

- `/var/parallels/`*`dest_VM_Name`*`.pvm` storing the files of a cloned virtual machine (where *`dest_VM_Name`* denotes the name of the resulting virtual machine). For example, for the `ClonedVM` virtual machine, this directory is `/var/parallels/ClonedVM.pvm`. To store the files of the `ClonedVM` virtual machine in a custom directory, you can run the following command:

```
# prlctl clone MyVM --name ClonedVM --dst /customVMs
```

  In this case all virtual machine files will be placed to the `/customVMs` directory. Notice that the specified directory must exist on the server; otherwise, the command will fail.

- `/vz/private/<`*`dest_CTID`*`>` defining the Container private area (where *<dest_CTID>* denotes the ID of the resulting Container). For Container 111, this path is `/vz/private/111`. To define a custom private area path for Container 111, you can execute the following command:

```
# prlctl clone 101 --name 111 --dst /vz/private/customCTs
```

Note: The default `/var/parallels` and `/vz/private` are valid for servers that do not participate in Parallels Cloud Storage clusters. If your server is part of a cluster, consult the *Parallels Cloud Storage Administrator's Guide* for information on the default directories for storing virtual machines and Containers.

Suspending Virtual Machines and Containers

Parallels Cloud Server allows you to suspend a running virtual machine and Container on the physical server by saving its current state to a special file. Later on, you can resume the virtual machine and Container and get it in the same state the virtual machine and Container was at the time of its suspending. Suspending your virtual machines and Containers may prove useful, for example, if you need to restart the physical server, but do not want to:

- quit the applications currently running in the virtual machine and Container
- spend much time on shutting down the guest operating system and then starting it again

You can use the `prlctl suspend` command to save the current state of a virtual machine and Container. For example, you can issue the following command to suspend the `MyVM` virtual machine:

```
# prlctl suspend MyVM
```

At any time, you can resume the `MyVM` virtual machine by executing the following command:

```
# prlctl resume MyVM
```

Once the restoration process is complete, any applications that were running in the `MyVM` virtual machine at the time of its suspending will be running again and the information content will be the same as it was when the virtual machine was suspended.

Running Commands in Virtual Machines and Containers

Parallels Cloud Server allows you to execute arbitrary commands inside virtual machines and Containers by running them on the physical server, i.e. without the need to log in to the respective virtual machine and Container. For example, this can be useful in these cases:

- If you do not know the virtual machine and Container login information, but need to run some diagnosis commands to verify that it is operational.
- If network access is absent for a virtual machine and Container.

In both these cases, you can use the `prlctl exec` command to run a command inside the respective virtual machine and Container. The session below illustrates the situation when you run the stopped SSH daemon inside a Linux virtual machine with the name of `My_Linux`:

```
# prlctl exec My_Linux /etc/init.d/sshd status
sshd is stopped
# prlctl exec My_Linux /etc/init.d/sshd start
Starting sshd:[OK]
# prlctl exec My_Linux /etc/init.d/sshd status
sshd (pid 26187) is running...
```

Notes:

1. You can use the `prlctl exec` command only inside virtual machines that have Parallels Tools installed.

2. The `prlctl exec` command is executed inside a virtual machine and Container from the `/` directory rather than from the `/root` one.

Deleting Virtual Machines and Containers

You can delete a virtual machine and Container that is not needed anymore using the `prlctl delete` command. Note that you cannot delete a running or mounted virtual machine and Container. The example below illustrates deleting the running Container 101:

```
# prlctl delete 101
Deleting Container private area: /vz/private/101
Container is currently running (stop first)
# prlctl stop 101
Stopping Container...
Container was stopped
Container is unmounted
# prlctl delete 101
Deleting Container private area: /vz/private/101
Container private area was deleted
```

Viewing Detailed Information About Virtual Machines and Containers

To view detailed information about a virtual machine or Container, you can use the `prlctl list -i` command. For example, the following command lists all information about the `MyVM` virtual machine:

```
# prlctl list -i MyVM
ID: {5c1fb1bb-4364-4b42-86b2-c584bdd2223b}
EnvID: 2075205468
Name: MyVM
Description:
State: running
OS: win-7
Uptime: 21:49:55 (since 2011-05-19 12:25:19)
Home: /var/parallels/MyVM.pvm/
Owner: root@.
Effective owner: owner
GuestTools: state=installed
```

```
Autostart: off
Autostop: shutdown
Boot order: hdd0 cdrom0 fdd0
Remote display: mode=off port=6500 address=0.0.0.0
Remote display state: stopped
Hardware:
  cpu 1 VT-x accl=high mode=32 ioprio=4 iolimit=0
  memory 1024Mb
  video 32Mb
  memory_quota auto
  fdd0 (+) real='/dev/fd0' state=disconnected
  hdd0 (+) sata:0 image='/var/parallels/MyVM.pvm/MyVM-0.hdd' 65536Mb
  cdrom0 (+) sata:1 real='D: ' state=disconnected
  parallel0 (+) real='/dev/lp0'
  usb (+)
  net0 (+) dev='vme7bb11f5c.0' network='Bridged' mac=001C427B68E3 card=e1000
Host Shared Folders: (+)
SmartMount: (-)
VirtualUsbMouse: state=disabled
Encrypted: no
Offline management: (-)
```

The following table describes the main options displayed by `prlctl list -i`.

Option	Description
`ID`	Virtual machine identifier. Usually, you use this ID, along with the virtual machine name, when performing an operation on the virtual machine.
`EnvID`	Kernel virtual machine identifier. This is the ID the kernel on the physical server uses to refer to a virtual machine when displaying some information on this virtual machine.
`Name`	Virtual machine name.
`Description`	Virtual machine description.
`State`	Virtual machine state.
`OS`	Guest operating system installed in a virtual machine.
`Uptime`	Time that shows for how long a virtual machine has been running since counter reset. **Note:** The uptime counter as well as count start date and time can be reset with the `prlctl reset-uptime` command.
`Home`	Directory storing virtual machine files.
`Guest Tools`	Shows whether Parallels Tools are installed in a virtual machine.
`Autostart`	Shows whether a virtual machine is automatically started when you turn on the physical server.
`Boot order`	Order in which the virtual machine devices are checked for an operating system.
`Hardware`	Devices available in a virtual machine.
`Offline management`	Denotes whether the offline management feature is enabled for the virtual machine, and if yes, lists the available offline services.

Note: The options `prlctl list` displays for Containers are similar to those for virtual machines.

Managing Virtual Machine and Container Backups

A regular backing up of existing virtual machines and Containers is essential for any system reliability. In Parallels Cloud Server, you can use the following utilities to back up and restore your virtual machines and Containers:

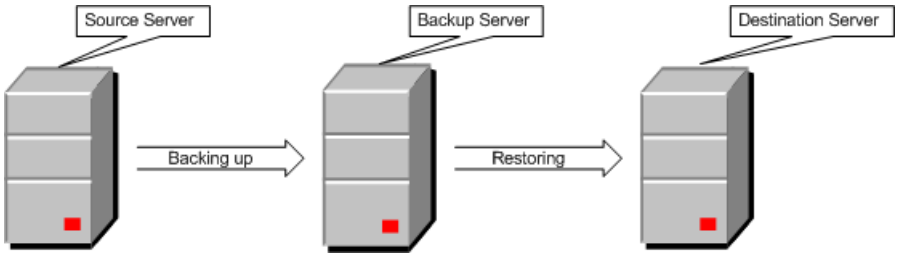
- `prlctl`
- `pbackup`
- `prestore`

Detailed information on these utilities is provided in the following subsections.

Backups Overview

Parallels Cloud Server backup utilities deal with three kinds of servers:

- *Source Server*. This is the server where virtual machines and Containers are hosted during their backing up.
- *Backup Server*. This is the server where virtual machine and Container backups are stored. A Backup Server can be any server running the Parallels Cloud Server software and having sufficient space for storing virtual machine and Container backups.
- *Destination Server*. This is the server where virtual machine and Container backups are restored.



These servers are singled out by their functionality only. In reality, one and the same physical server can perform two or even three functions. Usually, the Source and Destination Servers are represented by one and the same server because you will likely want the virtual machines and Containers you back up to be restored to their original server. However, setting up a dedicated Backup Server is recommended.

Creating Consistent Backups of Virtual Machines

Parallels Cloud Server allows you to back up both running and stopped virtual machines. However, to create a consistent backup of a running virtual machine, the virtual machine must meet the following requirements:

- Have Parallels Tools installed.
- Run one of the following operating systems:

  Windows operating systems

  - Windows Server 2012
  - Windows Server 2008 R2
  - Windows Server 2008
  - Windows Server 2003
  - Windows 8
  - Windows 7
  - Windows XP

  Linux operating systems

  - Red Hat Enterprise Linux 5 and 6
  - CentOS 5 and 6
  - Fedora 16 and 17
  - Debian 6
  - Ubuntu 11 and 12
  - SUSE Linux Enterprise Server 10
  - openSUSE 12

Using prlctl backup and prlctl restore

This section describes how to perform the basic backup-related operations using the `prlctl` utility.

Creating a Virtual Machine and Container Backup

You can use the `prlctl backup` command to back up virtual machines and Containers. This command is executed on the Source Server and can store the created virtual machine and Container backup on both the Source and Backup Servers. When creating a backup on the Source Server, you only need to specify the name of the virtual machine and Container to back up. For example, you can execute the following command to back up the `MyVM` virtual machine and store its backup archive on the Source Server:

```
# prlctl backup MyVM
Backing up the VM MyVM
...
The virtual machine has been successfully backed up with backup ID {746dba2a-3b10-4ced-
9dd6-76a2blcl4a69}
```

The command output informs you that the virtual machine backup has been successfully created and assigned ID `746dba2a-3b10-4ced-9dd6-76a2blcl4a69`. You can use this ID when managing the backup archive (e.g., remove the backup).

At the same time, you can run the following command to back up the `MyVM` virtual machine and store its backup archive on the Backup Server with the IP address of `129.129.10.10`:

```
# prlctl backup MyVM -s root:1qaz2wsx@129.129.10.10
```

`root:1qaz2wsx` before the Destination Server IP address denotes the root credentials used to log in to this server. If you do not specify these credentials, you will be asked to do so during the command execution.

By default, all newly created backups are placed to the following directories:

- `/vz/backups` for Containers
- `/var/parallels/backups` for virtual machines

For Containers, you can configure the default directory by changing the value of the `BACKUP_DIR` parameter in the `/etc/vzbackup.conf` configuration file. In turn, to set the default backup directory for virtual machines, you should use the `prlsrvctl set` command.

Notes:

1. A Backup Server can be any server running the Parallels Cloud Server software and having sufficient space for storing virtual machine and Container backups.

2. For more information on the options you can pass to `prlctl backup`, refer to the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Listing the Existing Backups

You can use the `prlctl backup-list` command to view the backups existing on the physical server. For example:

```
# prlctl backup-list
          ID  Backup_ID     Node     Date                Type Size
{c1dee22f...} {209d54a0...} test.com 2011-05-30 10:19:32 f    411566405
[The ID and Backup ID are reduced for better readability.]
```

This command lists the backups existing on the Source Server. If you want to list the backups on the Backup Server, you need to specify the IP address of this server.

The command output shows that currently only one backup exists on the Source Server. This backup was assigned the ID of `c1dee22f-8667-4870-9e11-278f1398eab0` (the full ID is skipped in the command output). The information on the backup is presented in the following table:

Column	Description
ID	The ID uniquely identifying the virtual machine and Container.
Backup ID	The ID assigned to the backup archive. You need to specify this ID when performing any backup-related operations.
Node	The hostname of the physical server storing the backup archive.
Date	The date and time when the backup archive was created.
Type	The backup type. Currently, you can create two types of backups: • A full backup indicated by f. • An incremental backup indicated by i and containing only the files changed since the previous full or incremental backup. This is the default backup type.
Size	The size of the backup archive, in bytes.

Removing a Virtual Machine and Container Backup

At any time, you can remove a backup that you do not need any more using the `prlctl backup-delete` command. To do this, you need to specify the ID of the backup to remove and the ID of the respective virtual machine and Container. If you do not know these IDs, use the `prlctl backup-list` and check the `ID` and `Backup ID` columns. For example:

```
# prlctl backup-list
          ID  Backup_ID     Node     Date                Type Size
{c1dee22f...} {209d54a0...} test.com 2011-05-30 10:19:32 f    411566405
[The ID and Backup ID are reduced for better readability.]
# prlctl backup-delete c1dee22f-8667-4870-9e11-278f1398eab0 -t 209d54a0-e3b8-4a03-9ca8-d4cc7a2a27ca
Delete the VM backup
The VM backup has been successfully removed.
```

You can also specify the virtual machine and Container name instead of its ID:

```
# prlctl backup-delete MyVM -t 209d54a0-e3b8-4a03-9ca8-d4cc7a2a27ca
```

If you have several backups of a particular virtual machine and Container and want to delete them all at once, indicate only the virtual machine and Container name or ID:

```
# prlctl backup-delete MyVM
```

This command removes all backups of the `MyVM` virtual machine from the local Backup Server. To remove backups stored remotely, you also need to specify the IP address of the remote Server:

```
# prlctl backup-delete MyVM -s root:1qaz2wsx@129.129.10.10
```

Restore a Virtual Machine and Container

To restore a backup of a virtual machine and Container, you can use the `prlctl restore` command. This command supports restoring backups to the Source Server only. For example, to restore a backup of the `MyVM` virtual machine stored on the Backup Server with the IP address of `10.10.100.1`, you can run this command on the Source Node:

```
# prlctl restore MyVM -s root:1qaz2wsx@10.10.100.1
```

If you have two or more backups of the MyVM virtual machine, the latest backup is restored. If you want to restore a particular virtual machine and Container backup, you need to specify the ID of this backup. You can use the prlctl backup-list command to list the existing backups and the IDs assigned to them:

```
# prlctl backup-list -s root:1qaz2wsx@10.10.100.1
          ID  Backup_ID     Node     Date                Type Size
{c1dee22f...} {209d54a0...} test.com 2011-05-30 10:19:32 i    11566405
{c1dee22f...} {24a3011c...} test.com 2011-05-21 11:12:35 f    356798701
[The ID and Backup ID are reduced for better readability.]
```

You can now indicate the desired ID after the -t option to tell prlctl backup to restore this particular backup. For example, to restore the backup for the virtual machine with the ID of c1dee22f-8667-4870-9e11-278f1398eab0 that was created on the 21st of May, you can execute this command:

```
# prlctl restore -t {24a3011c-8667-4870-9e11-278f1398eab0} -s root:1qaz2wsx@10.10.100.1
```

Note: Virtual machines created on servers running Parallels Cloud Server 6.0 cannot be restored on servers with Parallels Server 4 Bare Metal.

Using pbackup and prestore

Along with prlctl, you can use the following utilities to create and manage backups of your virtual machines and Containers:

- pbackup. This utility is used to create backups of individual virtual machines and Containers or entire Parallels servers.
- prestore. This utility is used to manage the existing backups of virtual machines and Containers.

Backing Up Virtual Machines and Containers

The pbackup utility is run on the Backup Server connecting via SSH to the Parallels server and backing up one or more virtual machines and Containers on this server. By default, the created Container backups are placed to the /vz/backups directory and the backups of virtual machines are put to the /var/parallels/backups directory. You can change the default backup directories

- for Containers, by editing the /etc/vzbackup.conf file
- for virtual machines, by using the --backup-path option of the prlsrvctl set command

Let us assume that you want to back up the entire Parallels server (that is, all virtual machines and Containers on this server) with the test.com hostname. In this case, you can run the following command on the Backup Server:

```
# pbackup test.com
```

During the command execution, you will be asked to provide the test.com credentials. After doing so, the command will back up all virtual machines and Containers on the test.com and put

- all backed up Containers to the Backup Server
- all backed up virtual machines to the Source Server

To save the backed up virtual machines also on the Backup Server, you should additionally specify the `-n` option. This option is used to indicate the IP address or hostname of the Backup Server and its credentials:

```
# pbackup -n root:7ujn6yhb@192.168.10.199 test.com
```

If you wish to back up not all, but specific virtual machines and Containers from the specified server, use the `-e` or `-x` switches (to include or exclude the specified virtual machines and Containers, respectively). For example:

```
# pbackup -n root:7ujn6yhb@192.168.10.199 test.com -e 101 MyVM
```

In this session, only Container 101 and the `MyVM` virtual machine residing on the Source Server with the `test.com` hostname will be included in the backup, and their backups will be stored on the Backup Server.

Notes:

1. A Backup Server can be any server running the Parallels Cloud Server software and having sufficient space for storing virtual machine and Container backups.

2. For the full list of configuration parameters and command line options for `pbackup`, consult the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Restoring Backups

To restore any individual virtual machines and Containers or entire Parallels servers, you may want to view first the information about them. This can be done using the `prestore -l` command:

```
# prestore -l -n test.com test.com
root@test.com's password:
...
Backups for node test.com:
           ID Backup_ID     Node     Date                Type Size
          101 2011-05-...   test.com 2012-05-30 09:42:19 f    18721280
{cd91b90b...} {4ef87485...} test.com 2012-05-16 17:15:47 f    92617398
[The ID and Backup ID are reduced for better readability.]
```

The command output shows that currently only two backups exist for the `test.com` server on the Backup Server. If you omit the `-n test.com` option, the command will list:

- all Container backups for the `test.com` server stored on the Backup Server
- all virtual machine backups for the `test.com` server stored on the `test.com` server

The information on the backups is presented in the following table:

Column	Description
`ID`	The ID uniquely identifying the virtual machine and Container.
`Backup ID`	The ID assigned to the backup archive. You need to specify this ID when performing any backup-related operations.
`Node`	The hostname of the Source Server.
`Date`	The date and time when the backup archive was created.
`Type`	The backup type. Currently, you can create two types of backups: • A full backup indicated by `f`. • An incremental backup indicated by `i` and containing only the files changed since the previous full or incremental backup. This is the default backup type.
`Size`	The size of the backup archive, in bytes.

To restore Container 101 and the `{cd91b90b-469d-42c6-acf4-fefee09cfa61}` virtual machine, run this command:

```
# prestore -n test.com -e 101 {cd91b90b-469d-42c6-acf4-fefee09cfa61}
```

This command will restore the Container and the virtual machine to their Source Server.

You can also use the `-d` option to restore Container 101 to a Parallels server other than the Source Node. For example, this command

```
# prestore -d 192.168.10.199 test.com -e 101
```

restores Container 101 to the Destination Server with IP address `192.168.10.199`. If you want to restore all Containers backups for the `test.com` Parallels server, just skip the `-e` option.

Notes:

1. The current version of Parallels Cloud Server supports restoring virtual machines to the Source Server only.

2. The `prestore` utility can also manage (list, restore, etc.) backups created using the `prlctl backup` command. However, you are highly recommended to use the same utility (either `prlctl` or `prestore`) during the life cycle of a particular backup.

3. For the full list of command line options for `prestore`, refer to the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Configuring Passwordless Access to the Source Node

You need to provide the Source Node credentials each time you execute the `pbackup` and `prestore` commands. However, you can allow these utilities to log in to the Source Node without having to enter the `root` password. To do this, you need to provide each Source Node with authorized public SSH RSA keys:

1 Log in to the Backup Server as `root`, and generate a pair of SSH keys - public and private:

```
# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
c6:19:a8:2c:67:31:15:e6:30:23:2b:8a:b0:63:77:8f root@dhcp-130.parallels.com
```

Note that you must leave an empty passphrase in the above procedure. The private key is saved by default in `/root/.ssh/id_rsa`, and the public key is saved in `/root/.ssh/id_rsa.pub`.

2 Transfer your public key to the `/root/.ssh` directory on each Source Node (use some intermediary name for the file not to overwrite the corresponding file on the Source Node):

```
# scp /root/.ssh/id_rsa.pub root@dhcp-129.parallels.com:/root/.ssh/temp_name
The authenticity of host 'dhcp-129.parallels.com (192.168.1.129)' can't be established.
RSA key fingerprint is 01:fc:b6:e9:26:40:1f:1a:41:5f:7a:fb:cf:14:51.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'dhcp-129.parallels.com,192.168.1.129' (RSA) to the list of
known hosts.
root@dhcp-129.parallels.com's password:
id_rsa.pub       100% |****************************|   235      00:00
```

3 Add the contents of the transferred file to the `authorized_keys` file in this very directory on the Source Node. To do this, log in to the Source Node, change to the `/root/.ssh` directory, and issue the following command:

```
# cat temp_name >> authorized_keys
```

Now the `pbackup/prestore` utilities should be able to log in to the Source Node as `root` without having to provide the `root` password.

Managing Templates

A template in Parallels Cloud Server is a pre-configured virtual machine or Container that can be easily and quickly deployed into a fully functional virtual machine or Container. Like any normal virtual machine or Container, a template contains hardware (virtual disks, peripheral devices) and the operating system. It can also have additional software installed. In fact, the only main difference between a virtual machine or Container and a template is that the latter cannot be started.

In Parallels Cloud Server, you can perform the following operations on templates:

- create a new template
- list the existing templates
- create a virtual machine or Container from a template

These operations are described in the following subsections in detail.

Creating Templates

In Parallels Cloud Server, you can create a template using the `prlctl clone` utility. Making a template may prove useful if you need to create several virtual machine or Container with the same configuration. In this case, your steps can be as follows:

1. You create a virtual machine or Container with the required configuration.
2. You make a template on the basis of the created virtual machine or Container.
3. You use the template to create as many virtual machine or Container as necessary.

Let us assume that you want to create a template of the `MyVM` virtual machine. To do this, you can run the following command:

```
# prlctl clone MyVM --name template1 --template
```

This command clones the `MyVM` virtual machine and saves it as the `template1` template. After the template has been successfully created, you can use it for creating new virtual machines.

Listing Templates

Sometimes, you may need to get an overview of the templates available on your Parallels server. For example, this may be necessary if you plan to create a virtual machine or Container from a specific template, but do not remember its exact name. In this case, you can use the `prlctl list` command to list all templates on the Parallels server and find the necessary one:

```
# prlctl list -t
{4ad11c28-9f0e-4086-84ea-9c0487644026}  win-2008        template1
{64bd8fea-6047-45bb-a144-7d4bba49c849}  rhel            template3
{6d3c9d6f-921a-484d-9772-bc7096f68df1}  win-2008        template2
```

In this example, 3 templates exist on the server. The information on these templates is presented in the form of a table with the following columns (from left to right): the template ID, the operating system contained in the template, and the template name.

Deploying Templates

To convert a template into a virtual machine or Container, use the `--ostemplate` option of the `prlctl create` command. For example, to convert the `template1` template to a virtual machine with the `ConvertedVM` name, you can run this command:

```
# prlctl create ConvertedVM --ostemplate template1
```

To check that the `ConvertedVM` virtual machine has been successfully created, use the `prlctl list -a` command:

```
# prlctl list -a
STATUS       IP_ADDR         NAME
running      10.12.12.101    111
stopped      10.12.12.34     Converted_VM
running      10.30.17.149    Windows7
```

The template itself is left intact and can be used for creating other virtual machines:

```
# prlctl list -t
{4ad11c28-9f0e-4086-84ea-9c0487644026}  win-2008        template1
{64bd8fea-6047-45bb-a144-7d4bba49c849}  rhel            template2
```

Managing Snapshots

In Parallels Cloud Server, you can save the current state of a virtual machine or Container by creating a snapshot. You can then continue working in your virtual machine or Container and return to the saved state any time you wish. Snapshots may be useful in the following cases:

- Configuring applications with a lot of settings. You may wish to check how settings work before applying them to the application. So, before you start experimenting, you create a snapshot.
- Participating in large-scale development projects. You may wish to mark development milestones by creating a snapshot for each. If anything goes wrong, you can easily revert to the previous milestone and resume the development.

In Parallels Cloud Server, you can create, list, revert to, and delete snapshots. These operations are described in the following subsections.

Creating Snapshots

To create a snapshot of a virtual machine or Container, use the `prlctl snapshot` command.

Creating Virtual Machine Snapshots

To create a snapshot of the virtual machine `MyVM`, do the following:

```
# prlctl snapshot MyVM
...
The snapshot with ID {12w32198-3e30-936e-a0bbc104bd20} has been successfully created.
```

A newly created snapshot is saved to the `/vz/`*`VM_Name`*`.pvm/Snapshots/`*`Snapshot_ID`*`.pvs` file, where *`VM_Name`* is the corresponding virtual machine name and *`Snapshot_ID`* is a unique snapshot ID. In the above example, the snapshot with ID `{12w32198-3e30-936e-a0bbc104bd20}` is saved to the file `/vz/MyVM/Snapshots/{12w32198-3e30-936e-a0bbc104bd20}.pvs`.

```
# ls /vz/MyVM.pvm/Snapshots/
{063615fa-f2a0-4c14-92d4-4c935df15840}.pvc
```

Snapshot IDs are needed to switch to and delete snapshots.

When creating a snapshot, you can also set its name and description:

```
# prlctl snapshot MyVM -n Clean_System -d "This snapshot was created right after
installing Windows XP."
...
The snapshot with ID {0i8798uy-1eo0-786d-nn9ic106b9ik} has been successfully created.
```

You can then view the set name and description in the `/vz/MyVM/Snapshots.xml` file.

Creating Container Snapshots

To create a snapshot of Container 101, do the following:

```
# prlctl snapshot 101
...
The snapshot with ID {08ddd014-7d57-4b19-9a82-15940f38e7f0} has been successfully
created.
```

A newly created snapshot is saved to the `/vz/private/<CT_ID>/dump/<snapshot_ID>` file, where *<CT_ID>* is the Container ID and *<snapshot_ID>* is a snapshot ID. In the above example above, the snapshot with ID `{08ddd014-7d57-4b19-9a82-15940f38e7f0}` is saved to the file `/vz/private/101/dump/{08ddd014-7d57-4b19-9a82-15940f38e7f0}`.

```
# ls /vz/private/101/dump
{08ddd014-7d57-4b19-9a82-15940f38e7f0}
```

Snapshot IDs are needed to switch to and delete snapshots.

When creating a snapshot, you can also set its name and description:

```
# prlctl snapshot 101 --n Clean_System --d "This snapshot was created right after
installing Windows XP."
...
The snapshot with ID {e78bb2b8-7a99-4c8b-ab9a-491a47648c44} has been successfully
created.
```

The set name and description are stored in the `/vz/private/<CT_ID>/Snapshots.xml` file.

Snapshot Branching

Snapshot branches can be useful for working with, testing or comparing similar configurations. A snapshot branch is created when you do the following:

1. Create several snapshots.
2. Revert to one of the snapshots.
3. Make changes to the virtual machine or Container.
4. Create a snapshot.

In this case, the newly created snapshot will start a new branch based on the snapshot from **Step 2**.

Restrictions and Recommendations

- Virtual machine and snapshot names and snapshot descriptions containing spaces must be enclosed in quotation marks (e.g., "Windows XP") when supplying them to the `prlctl` command.
- Before creating a snapshot, it is recommended that you finish any installations, downloads, and stop writing to external devices. You should also complete or cancel any transactions

performed via the virtual machine in external databases.

Listing Snapshots

To list all snapshots of a particular virtual machine or Container, use the `prlctl snapshot-list` command. For example, to check all current snapshots of the `MyVM` virtual machine, run this command:

```
# prlctl snapshot-list MyVM
PARENT_SNAPSHOT_ID                        SNAPSHOT_ID
                                          {989f3415-3e30-4494-936e-a0bbc104bd20}
{989f3415-3e30-4494-936e-a0bbc104bd20} *{063615fa-f2a0-4c14-92d4-4c935df15840}
```

This command shows that the `MyVM` virtual machine has two snapshots. The snapshot with ID `{063615fa-f2a0-4c14-92d4-4c935df15840}` is based on the snapshot with ID `{989f3415-3e30-4494-936e-a0bbc104bd20}`, i.e. the former is a child of the latter. The asterisk marks the current snapshot.

To view the relationships between snapshots, use the `-t` option:

```
# prlctl snapshot-list MyVM -t
_{989f3415-3e30-4494-936e-a0bbc104bd20}_{063615fa-f2a0-4c14-92d4-
4c935df15840}*{712305b0-3742-4ecc-9ef1-9f1e345d0ab8}
```

The command output shows you that currently two branches exist for the `MyVM` virtual machine. The snapshot with ID `{989f3415-3e30-4494-936e-a0bbc104bd20}` is the base for these branches.

To get detailed information on a particular snapshot, use the `-i` option with the snapshot ID:

```
# prlctl snapshot-list MyVM -i {063615fa-f2a0-4c14-92d4-4c935df15840}
ID: {063615fa-f2a0-4c14-92d4-4c935df15840}
Name: Clean_System
Date: 2012-07-22 22:39:06
Current: yes
State: poweroff
Description: <![CDATA[This snapshot was created right after installing Windows 7]]>
```

The `prlctl snapshot-list` command with the `-i` option displays the following information about snapshots:

Field	Description
`ID`	ID assigned to the snapshot.
`Name`	Name assigned to the snapshot.
`Date`	Date and time when the snapshot was created.
`Current`	Denotes that this is the current snapshot of the virtual machine.
`State`	State the virtual machine was in at the time you took the snapshot.
`Description`	The description set for the snapshot.

Reverting to Snapshots

To revert to a snapshot, use the `prlctl snapshot-switch` command. When you revert to a snapshot, the current state of the virtual machine or Container is discarded, and all changes made to the system since the previous snapshot are lost. So, before reverting, you may want to save the current state by creating a new snapshot (see **Creating Snapshots (p. 46)**).

The `prlctl snapshot-switch` command requires the virtual machine or Container ID and the snapshot ID as arguments, for example:

```
# prlctl snapshot-switch MyVM --id {cedbc4eb-dee7-42e2-9674-89d1d7331a2d}
```

In this example, you revert to the snapshot `{cedbc4eb-dee7-42e2-9674-89d1d7331a2d}` for the virtual machine `MyVM`.

Deleting Snapshots

To delete unneeded snapshots of virtual machines or Containers, use the `prlctl snapshot-delete` command. For example:

```
# prlctl snapshot-delete MyVM --id {903c12ea-f6e6-437a-a2f0-a1d02eed4f7e}
```

When you delete a parent snapshot, child snapshots are not deleted, and the information from the former is merged into the latter.

Migrating Virtual Machines and Containers

The Parallels physical server is the system with higher availability requirements in comparison with a typical system. If you are running your company mail server, file server, and web server in different virtual machines and Containers on one and the same physical server, then shutting it down for hardware upgrade will make all these services unavailable at once. To facilitate hardware upgrades and load balancing between several Parallels servers, the Parallels Cloud Server software provides you with the ability to migrate virtual machines and Containers from one physical server to another.

Parallels Cloud Server is shipped with a special utility—`pmigrate`—allowing you to perform different types of migration. Using this utility, you can migrate

- Containers from one physical server to another
- Parallels virtual machines from one physical server to another
- a Container to a Parallels virtual machine
- a Parallels virtual machine to a Container
- a physical server to a virtual machine and Container
- Xen virtual machines to Parallels virtual machines

All these operations are described in the following subsections.

General Migration Requirements

Before deciding on the type of migration to perform, make sure that the source computer (i.e. the computer that you will migrate or that stores the virtual machine and Container before its migration) and the destination computer (i.e. the computer that runs Parallels Cloud Server and that will host the resulting virtual machine and Container) meet the requirements below.

Source computer requirements

The source computer can be a physical computer, a virtual machine, or a Container. The software requirements for source computers are given in the following table:

Operating System	Physical Computers	Virtual Machines	Containers
Windows			
Windows 8 (x32, x64)	+	+	-
Windows 7 (x32, x64)	+	+	-
Windows Server 2012 (x64)	+	+	-
Windows Server 2008 R2 (x64)	+	+	-
Windows Server 2003 (x32, x64)	+	+	+
Windows Server 2008 (x32, x64)	+	+	+
Windows XP (x32, x64)	+	+	-
Linux			
Red Hat Enterprise Linux 6 (x32, x64)	+	+	+
Red Hat Enterprise Linux 5 (x32, x64)	+	+	+
CentOS 6 (x32, x64)	+	+	+
CentOS 5 (x32, x64)	+	+	+
Fedora 17 (x32, x64)	+	+	+
Fedora 16 (x32, x64)	+	+	+
SUSE Linux Enterprise Server 11 (x86, x64)	+	-	+
SUSE Linux Enterprise Server 10 (x32, x64)	+	+	-
openSUSE 12.2 (x86, x64	+	+	-
openSUSE 12.1 (x86, x64)	+	-	+
openSUSE 11.4 (x86, x64)	+	-	+
Debian GNU/Linux 6 (x32, x64)	+	+	+
Ubuntu 12.10 (x86, x64)	+	+	+
Ubuntu 12.04 (x86, x64)	+	-	+
Ubuntu 11.10 (x86, x64)	+	+	+

Ubuntu Linux 10.04 Server (x32, x64)	+	+	+
Ubuntu Server 12.04 (x86, x64)	+	+	-
Ubuntu Server 10.04 (x86, x64)	+	+	-

Destination server requirements

The destination server must meet the following requirements:

- Has enough hard disk space to store the resulting virtual machine and Container.
- Has enough memory and CPU power to run the resulting virtual machine and Container.
- Has a stable network connection with the source server.

Migrating Virtual Machines and Containers Between Parallels Servers

In Parallels Cloud Server, you can choose one of the following ways to migrate virtual machines and Containers:

- Migrating virtual machines and Containers using the standard migration technology.
- Migrating virtual machines and Containers using the zero-downtime migration technology.

Both ways of migrating virtual machines and Containers are described in the following subsections in detail.

Standard Migration

Using the standard migration technology, you can move

- stopped, running, and suspended Containers
- stopped and suspended virtual machines

Standard migration includes copying all files of a virtual machine or Container from one Parallels server to another and does not differ from copying a number of files from one server to another over the network. For a running Container, the migration procedure is a bit more complicated and is described below:

1. Once you start the migration, all Container data are copied to the destination server. During this time, the Container on the source server continues running.
2. The Container on the source server is stopped.
3. The Container data copied to the destination server are compared with those on the source server, and if any files were changed during the first migration step, they are copied to the destination server again and rewrite the outdated versions.
4. The Container on the destination server is started.

There is a short downtime needed to stop the Container on the source server, copy the changed data to the destination server, and start the Container on the destination server.

Migrating Containers

The following session moves Container 101 from the source server to the destination server `ts7.test.com`:

```
# pmigrate c 101 c root:XXXXXXXXX@ts7.test.com/101
```

The `c` option in the command above tells `pmigrate` that you are moving a Container to a Container. If you do not indicate the credentials to log in to the destination server, you will need to do so during the migration.

Important! For the command to be successful, a direct SSH connection (on port 22) must be allowed between the source and destination servers.

By default, after the migration process is completed, the Container private area and configuration file are renamed on the source server by receiving the `.migrated` suffix. However, if you want the Container private area on the source server to be removed after the successful Container migration, you can override the default `pmigrate` behavior by changing the value of the `REMOVEMIGRATED` variable in the Parallels Cloud Server global configuration file (`/etc/vz/vz.conf`) to `yes` or by using the `-r yes` switch with the `pmigrate` command.

Migrating Virtual Machines

In turn, to migrate a virtual machine from the source server to `ts7.test.com`, you need just to specify `v` instead of `c` and the name of the resulting virtual machine instead of Container ID 101:

```
# pmigrate v MyVM v root:XXXXXXXXX@ts7.test.com/MyVM
```

This command moves the `MyVM` virtual machine from the local server to the destination server `ts7.test.com`. Once the migration is complete, the original virtual machine is removed from the source server. However, you can use the `--keep-src` option to leave the original virtual machine intact.

For virtual machines, `pmigrate` also supports the migration from a remote Parallels server to the local one:

```
# pmigrate v ts7.test.com/MyVM v localhost
```

This command moves the `MyVM` virtual machine from the `ts7.test.com` server to the local server.

Note: For more information on options that you can pass to `pmigrate`, refer to the *Parallels Cloud Server 6.0 Command Line Reference*.

Zero-Downtime Migration

Using the zero-downtime migration technology, you can migrate paused and running virtual machines and running Containers from one Parallels server to another with zero downtime. The zero-downtime migration technology has the following main advantages over the standard one:

- The migration time is greatly reduced. In fact, the migration eliminates the service outage or interruption for end users.
- The process of migrating a virtual machine or Container to another Parallels server is transparent for you and the Container applications and network connections. This means that no modifications of system characteristics and operational procedures inside the Container are performed on the source and destination servers.
- The virtual machine or Container is restored on the destination server in the same state as it was at the beginning of the migration.
- You can move virtual machines and Containers running applications that you do not want to be rebooted during the migration.

Note: Zero-downtime migration is not supported for virtual machines and Containers that have open sessions established with the `prlctl enter` command.

Migration requirements and restrictions

When performing a zero-downtime migration, take into account the requirements and restrictions below:

- Before performing zero-downtime migration, it is recommended to synchronize the system time on the source and destination servers, for example, by means of NTP (http://www.ntp.org). The reason for this recommendation is that some processes running in virtual machines and Containers might rely on the system time being monotonic and thus might behave unpredictably if they see an abrupt step forward or backward in the time once they find themselves on the new server with different system clock parameters.
- Your network must support data transfer rates of at least 1 Gb/s.
- The source and destination servers must belong to the same subnetwork.
- The CPUs on the source and destination servers must be manufactured by the same vendor, and the CPU capabilities on the destination server must be the same or exceed those on the source server.
- virtual machine and Container disks can be located on local disks, shared NFS and GFS2 storages, and ISCSI raw devices.

Migration process overview

The process of migrating virtual machines and Containers using the zero-downtime migration technology includes the following main steps:

1. Once you start the migration, Parallels Cloud Server checks whether the destination server meets all migration requirements and the virtual machine or Container can be migrated to this server.
2. All virtual memory and disks of the virtual machine or Container are migrated to the destination server.
3. The virtual machine or Container on the source server is suspended.

4 The changed memory pages and virtual disk blocks, if any, are migrated to the destination server.

5 The virtual machine or Container is resumed on the destination server.

The virtual machine or Container continues running during steps 1 and 2 and is not available to the end user during steps 3-5. But since the amount of memory pages and virtual disk blocks changed during step 2 is small, the service outage time for the end user is almost imperceptible.

Migrating virtual machines and Containers

Depending on whether you are migrating a virtual machine or Container, the command-line options you pass to the `pmigrate` slightly differ. For example, you can migrate Container 101 from the local server to the destination server `destserver.com` by executing the following command on the local server:

```
# pmigrate c 101 c --online destserver.com
Enter password:
Connection to destination server (192.168.1.57) is successfully established
...
Successfully completed
```

At the same time, to migrate the `MyVM` virtual machine to the same destination server `destserver.com`, you can run this command on the local server:

```
# pmigrate v MyVM v destserver.com
Migrate the VM MyVM to test.com
...
The VM has been successfully migrated.
```

As you can see, to migrate a virtual machine, you skip the `--online` option and use the `v` option to specify that you are migrating a virtual machine.

Notes:

1. For more information on options you can use with the `pmigrate` utility, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

2. After migration, the moved virtual machine may not be accessible over the network for several minutes due to latencies in the network equipment reconfiguration (for example, when switches need to update their dynamic VLAN membership tables).

Migrating Containers to Virtual Machines

The `pmigrate` utility allows you to migrate Containers to virtual machines. The source server, i.e. the server where the Container resides before its migration, can be one of the following:

- a local server running Parallels Cloud Server
- a remote server running Parallels Cloud Server
- a remote server running Parallels Virtuozzo Containers

Currently, the destination server, i.e. the server where the resulting virtual machine will be created, can be only a local server with Parallels Cloud Server.

The process of migrating a Container to a virtual machine differs depending on whether the server where the Container resides is running the Parallels Cloud Server or Parallels Virtuozzo Containers software.

Migrating Containers

You can use the `pmigrate` utility to migrate Containers that reside on both local and remote servers running Parallels Cloud Server. When migrating a Container from a local server, you only need to specify the Container ID and the name of the resulting virtual machine. For example, the following command migrates Container 101 to the `VM_101` virtual machine on the same Parallels server:

```
# pmigrate c 101 v VM_101
```

The resulting virtual machine will be put to the `/var/parallels` directory on the destination server.

If you want to migrate a Container from a remote Parallels server, you should additionally indicate the source server IP address and the credentials of the root user on this server:

```
# pmigrate c root:8uhytv4@192.168.12.12/101 v VM_101
```

This command migrates Container 101 residing on the Parallels server with the IP address of `192.168.12.12` to the `VM_101` virtual machine on the local server. If you do not specify the root credentials on the source server, you will be asked to do so during the command execution.

Migrating Containers from Servers with Parallels Virtuozzo Containers

You can use the `pmigrate` utility to migrate Containers that reside on remote servers running the following versions of the Parallels Containers software:

- Parallels Virtuozzo Containers 4.0 for Linux with update TU-4.0.0-464 or higher
- Parallels Virtuozzo Containers 4.6 for Linux
- Parallels Virtuozzo Containers 4.7 for Linux
- Parallels Virtuozzo Containers 4.5 for Windows
- Parallels Virtuozzo Containers 4.6 for Windows

Moving a Container from a remote Parallels Containers server to a virtual machine on the local server with Parallels Cloud Server involves completing the following steps:

1. Installing the Parallels agent on the server with Parallels Containers.
2. Running the `pmigrate` utility on the destination server to migrate the Container.

Installing the Parallels for Containers Agent

First, you must install the Parallels agent on the source Parallels Containers server. During migration, the agent collects essential information on the Container to be moved and transfers it to the `pmigrate` utility on the destination server. To install the Parallels agent, do the following:

1. Log in to the source Parallels Containers server as a user with administrative rights.
2. Copy the Parallels agent installation file to the source server. The installation file is located in the `/usr/share/pmigrate/tools` directory on the server with Parallels Cloud Server:
   - `parallels-transporter-for-containers-`*`XXXX`*`.run`. Use this file to install the Parallels agent on servers running Parallels Virtuozzo Containers 4.0, 4.6, or 4.7 for Linux.
   - `ParallelsTransporterForContainers-parallels-`*`XXXX`*`.exe`. Use this file to install the Parallels agent on servers running Parallels Virtuozzo Containers 4.5 or 4.6 for Windows.
3. Execute the installation file on the source server.
4. Follow the instructions of the wizard to install the Parallels agent.

Migrating the Container

Once the Parallels agent is installed, you can use the `pmigrate` utility to move a Container to a virtual machine. For example, you can run the following command on the destination server to migrate Container 101 from the remote server with IP address `192.168.12.12` to the `VM_101` virtual machine:

```
# pmigrate c root:8uhytv4@192.168.12.12/101 v VM_101
Connecting to local agent...
Querying configuration...
Migrating...
Operation progress 100%
Registering VM...
PVC to VM /var/parallels/VM_101.pvm/config.pvs migration succeeded.
```

The resulting virtual machine will be put to the `/var/parallels` directory on the destination server. If you do not specify the administrative credentials on the source server (for `root` on Linux servers and `Administrator` on Windows servers), you will be asked to do so during the command execution.

Migrating Physical Computers to Virtual Machines and Containers

You can use the `pmigrate` utility to migrate a standalone physical computer to a virtual machine or Container. The procedure includes copying the entire contents of a physical computer, including all files, directories, quota limits, settings, and so on, to a virtual machine or Container on a Parallels server. Migration of a computer produces its exact copy in the form of a virtual machine or Container, including operating system, assigned IP addresses, disk space, memory capacity, etc.

Migrating a physical computer to a virtual machine or Container involves completing the following steps:

1. Installing the Parallels agent on the source physical computer (required for migrating to virtual machines only).
2. Running the `pmigrate` utility on the destination server.

Installing Parallels Agent

The Parallels agent collects essential system data on the source physical computer and supplies it to the `pmigrate` utility on the destination Parallels server. To install the agent, do the following:

1. Make sure that the source physical computer meets the requirements for installing the agent. See **Requirements for Migrating to Virtual Machines** (p. 59) for details.
2. Log in to the source physical computer as administrator.
3. Copy the correct agent installation file to the source physical computer from the `/usr/share/pmigrate/tools` directory on the destination Parallels server:
   - For a Linux OS, choose `parallels-transporter-agent-`*`XXXX`*`.run`.
   - For a Windows OS, choose `ParallelsTransporterAgent-parallels-`*`XXXX`*`.exe`.
4. Run the installation file on the source physical computer and follow the wizard's instructions.
5. Restart the source physical computer to complete the installation.

Note: The Parallels agent is launched automatically after restart. You do not need to run in manually.

Migrating Source Physical Computer

Once the source physical computer is up and running, you can migrate it to a virtual machine or Container on the destination Parallels server. For example, to migrate a physical computer to a virtual machine, run the following command on the destination server:

```
# pmigrate h root:1qsde34rt@192.168.1.130 v MyVM
```

Where:

- `h` indicates that the migration source is a physical computer.
- `root:1qsde34rt@192.168.1.130` contains the source physical computer credentials and IP address.

  If you omit credentials, you will be asked to provide them during command execution.
- `v` indicates that the migration destination is a virtual machine.
- `MyVM` is the name of the resulting virtual machine on the destination Parallels server.

Once the procedure is complete, you will find the resulting virtual machine in the `/var/parallels` directory on the destination Parallels server.

To migrate the same physical computer to a Container, just replace `v` with `c` and `MyVM` with a Container ID (e.g., 101). For example:

```
# pmigrate h root:1qsde34rt@192.168.1.130 c 101
```

Notes:

1. Migrating physical computers running Windows operating systems to Containers is not supported.

2. Migrating physical computers with ReiserFS system volumes is not supported.

Requirements for Migrating to Containers

To avoid delays and problems when migrating a physical server to a Container, make sure that the following requirements are met:

- The Linux distribution installed on the source physical computer is supported by Parallels Cloud Server. To find that out, check the `/etc/vz/conf/dists` directory on the destination Parallels server and look for a corresponding `Linux_Distribution_Name-version.conf` configuration file (e.g., `redhat-5.conf`). If there is none, you can do one of the following:
  - Create a new distribution configuration file and place it in the `/etc/vz/conf/dists` directory on the destination Parallels server. For more details, see **Creating Configuration Files for New Linux Distribution** (p. 181).
  - Start migration without a configuration file. In this case `unknown.conf` from the `/etc/vz/conf/dists` directory will be used to configure the resulting Container. However, you will not be able to use standard Parallels Cloud Server utilities (e.g., `prlctl`) to perform main operations on the newly created Container (e.g., set its IP address or configure DNS parameters) and have to do that manually from inside the Container.
- `ssh` is installed on both the source physical computer and the destination Parallels server to provide secure encrypted and authenticated communication between the two. You can check if the `ssh` package is already installed by executing the `ssh -V` command.
- `rsync` is installed on the source physical computer to copy the source contents to the resulting Container. If `rsync` on the source physical computer is incompatible with that on the destination Parallels server, use the latter, which is located in the `/usr/local/share/vzlinmigrate` directory.

Migration Restrictions for Containers

Listed below are the limitations you should take into account when migrating physical computers to Containers.

- If the source physical computer has several IP addresses assigned to it, all of those are reassigned to the same `venet0` device on the destination Parallels server. This virtual network adapter is used to interconnect all Containers on a Parallels server and the server itself. After migration, you can create additional virtual network adapters inside the resulting Container and assign necessary IP addresses to network adapters as you see fit. For more details, see

Managing Adapters in Containers (p. 138).

- During migration, you may specify only one partition on the source physical computer that will be migrated to a Container together with all its quotas. All other partitions will be copied without their quota limits. Moreover, the quota limits of the selected partition will be applied to the entire Container after migration.
- While migrating a physical server running a Linux operating system with a security-enhanced (SE) Linux kernel, keep in mind that Parallels Cloud Server currently does not support SE Linux kernels, and the resulting Container will not support the SE features of the source.
- If any of your files and/or directories on the source physical computer have extended attributes associated with them, these attributes will be lost after migration.
- Raw devices on the source physical computer cannot and will not be migrated to the resulting Container on the destination Parallels server.
- Any applications bound to the MAC address of the source physical computer will not run inside the resulting Container after migration. In this case, you can do one of the following:
  - Obtain new licenses for applications that require them and install those anew inside the Container.
  - Try to reconfigure applications which do not require a license for working without being bound to a MAC address.
- If migration fails at a stage when `rsync` is transferring files and directories from the source physical computer to the resulting Container, the already copied files and directories will remain in the `/vz/private/<CT_ID>` directory on the destination Parallels server, probably occupying much disk space. You can keep this directory to speed up a new migration attempt, or you can delete it manually by using the `rm` utility.
- Migrating physical computers with system volumes formatted as ReiserFS is not supported.
- Migration of GUID Partition Table (GPT) disks is not supported.

Requirements for Migrating to Virtual Machines

Any physical computer to be migrated to a virtual machine must have the Parallels agent installed, for which the following requirements must be met.

Hardware Requirements

- 700 MHz or faster x86 or x64 processor (Intel or AMD).
- 256 MB or more RAM.
- 50 MB of free hard disk space for the Parallels agent.
- Ethernet or Wi-Fi network adapter.

Software Requirements

See **General Migration Requirements** (p. 50).

Additional Requirements for Migrating Servers with Parallels Cloud Server

To migrate a server running the Parallels Cloud Server software, you should first make sure that the `snapapi26` and `snumbd26` modules are not loaded on that server. To check this, you can use the following commands:

```
# lsmod | grep snapapi26
# lsmod | grep snumbd26
```

If necessary, unload the modules by running the `rmmod` command. You may need to stop the Parallels Cloud Server service first, as explained below:

1. Stop the service:

```
# /etc/init.d/vz stop
```

2. Unload the modules:

```
# rmmod snapapi26
# rmmod snumbd26
```

3. Start the service again:

```
# /etc/init.d/vz start
```

Once the modules are unloaded, proceed with migration.

Migration Restrictions for Virtual Machines

Listed below are the limitations you should take into account when migrating physical computers to virtual machines.

- Migrating Windows dynamic volumes and Linux logical volumes (LVM) is not supported.
- Migrating Windows software RAID configurations is not supported.
- Migrating GUID Partition Table (GPT) disks is not supported.
- Migrating physical computers with ReiserFS system volumes is not supported.

Migrating Virtual Machines to Containers

The process of migrating a virtual machine to a Container on the Parallels server is the same as migrating a physical computer to a Container. For example, you can execute the following command to move a virtual machine with the IP address of 192.168.1.130 to Container 101 on your Parallels server:

```
# pmigrate h root:1qsde34rt@192.168.1.130 c 101
```

You can omit the virtual machine credentials in the command above. In this case you will be asked to provide them during the command execution.

Notes:

1. For more information on migrating physical computers to Containers, see **Migrating Physical Computers to Virtual Machines and Containers** (p. 56).

2. The requirements a virtual machine must meet are the same as for migrating physical computers; they are described in **Requirements for Migrating to Containers** (p. 58).

Migrating Xen Virtual Machines

You can use the `pmigrate` utility to migrate Xen virtual machines to virtual machines on the Parallels server. Moving a Xen virtual machine to a Parallels virtual machine involves completing the following steps:

1. Installing the Parallels agent on the physical server hosting the Xen virtual machine.
2. Migrating the Xen virtual machine by running the `pmigrate` utility on the server.

Both steps are described below in detail.

Installing the Agent

Before you start migrating a Xen virtual machine, you need first to install the Parallels agent on the Xen server where the virtual machine is residing. To install the Parallels agent, do the following:

1. Log in to the Xen server as a user with administrative rights.
2. Copy the Parallels agent installation file to the Xen server. The installation file is located in the `/usr/share/pmigrate/tools` directory on the Parallels server and has the name `parallels-transporter-agent-parallels-en_US-`*`XXXX`*`.run`.
3. Execute the copied file, and follow the instructions to install the Parallels agent.
4. Start the Parallels agent:

```
# parallels-transporter-agent -c
```

Migrating the Xen virtual machine

Once the Parallels agent is running on the Xen server, you can migrate the Xen virtual machine. Let us assume the following:

- You want to migrate the `XenVM` virtual machine from the Xen server to the `MigratedVM` virtual machine on the Parallels server.
- `root:1qsde34rt@192.168.1.130` is the IP address and credentials of the Xen server where the `MigratedVM` virtual machine resides.

To migrate the `XenVM` virtual machine, you can run the following command:

```
# pmigrate x root:1qsde34rt@192.168.1.130/XenVM v MigratedVM
```

In this command, `x` denotes that you are migrating a Xen virtual machine, and `v` indicates that the Xen virtual machine is to be moved to a Parallels virtual machine. If you omit the credentials in the command above, you will be asked to provide them during the command execution. Once the migration is complete, you can find the resulting virtual machine in the `/var/parallels` directory on the Parallels server.

Note: You are recommended to check the settings of the migrated virtual machine (for example, memory and network settings) and, if necessary, configure them to meet your needs.

Troubleshooting the migration of paravirtualized Xen virtual machines

When migrating a paravirtualized Xen virtual machine, `pmigrate` first copies the whole of the virtual machine to the Parallels server and then replaces the paravirtualized kernel of the copied machine with a normal kernel from the corresponding Linux distribution. If it cannot replace the kernel, `pmigrate` displays an error but does not delete the virtual machine from the Parallels server. In this case, you can do the following:

- Remove the copied virtual machine from the Parallels server and try to migrate the virtual machine again.
- Configure the copied virtual machine on the Parallels server manually.

If you choose the second way, do the following:

1. Boot into the virtual machine in rescue mode using an ISO image of the Linux OS corresponding to the OS installed in the virtual machine.
2. Detect where on the disk the root partition is located, and mount it.
3. Detect all other partitions on the disk (`/boot`, `/usr`, and so on), and mount them to the corresponding directories on the root partition; also mount the `/proc` file system.
4. Install a normal Linux kernel (for example, from the ISO image you used to boot into the virtual machine). The normal kernel must be of the same architecture as the paravirtualized Xen kernel.
5. Create the `initrd` image for the normal kernel if you have not already done so when installing the kernel.
6. Configure the bootloader to load the normal kernel if you have not already done so when installing the kernel.
7. Configure the `/etc/inittab` file to start `getty` and `tty1-tty6`.
8. Unmount the partitions.
9. Restart the virtual machine, and boot into the normal kernel.

Performing Container-Specific Operations

This section provides the description of operations specific to Containers.

Reinstalling Containers

Reinstalling a Container may help if any required Container files have been inadvertently modified, replaced, or deleted, resulting in Container malfunction. You can reinstall a Container by using the `prlctl reinstall` command that creates a new Container private area from scratch according to its configuration file and relevant OS and application templates. For example:

```
# prlctl reinstall 101
```

Notes:

1. If any of the Container application templates cannot be added to the Container in a normal way, reinstallation will fail. This may happen, for example, if an application template was added to the Container using the `--force` option of the `vzpkgadd` or `vzpkg install` command (for more information on these commands, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*).

2. Currently, the `reinstall` command may not be supported by the `prlctl` utility. Use `vzctl` instead.

To keep the personal data from the old Container, the utility also copies the old private area contents to the `/vz/root/<CT_ID>/old` directory of the new private area (unless the `--skipbackup` option is given). This directory may be deleted after you copy the personal data where you need.

The `prlctl reinstall` command retains user credentials base, unless the `--resetpwdb` option is specified.

Customizing Container Reinstallation

The default reinstallation, as performed by the `prlctl reinstall` command, creates a new private area for the broken Container as if it were created by the `prlctl create` command and copies the private area of the broken Container to the `/old` directory in the new private area so that no file is lost. There is also a possibility of deleting the old private area altogether without copying or mounting it inside the new private area, which is done by means of the `--skipbackup` option. This way of reinstalling corrupted Containers might in certain cases not correspond exactly to your particular needs. It happens when you are accustomed to creating new Containers in some other way than just using the `prlctl create` command. For example, you may install additional software licenses into new Containers, or anything else. In this case you would naturally like to perform reinstallation in such a way so that the broken Container is reverted to its original state as determined by you, and not by the default behavior of the `prlctl create` command.

To customize reinstallation, you should write your own scripts determining what should be done with the Container when it is being reinstalled, and what should be configured inside the Container after it has been reinstalled. These scripts should be named `vps.reinstall` and `vps.configure`, respectively, and should be located in the `/etc/vz/conf` directory on the server. To facilitate your task of creating customized scripts, the Containers software is shipped with sample scripts that you may use as the basis of your own scripts.

When the `prlctl reinstall <CT_ID>` command is called, it searches for the `vps.reinstall` and `vps.configure` scripts and launches them consecutively. When the `vps.reinstall` script is launched, the following parameters are passed to it:

Option	Description
`--veid`	Container ID.
`--ve_private_tmp`	The path to the Container temporary private area. This path designates where a new private area is temporarily created for the Container. If the script runs successfully, this private area is mounted to the path of the original private area after the script has finished.
`--ve_private`	The path to the Container original private area.

You may use these parameters within your `vps.reinstall` script.

If the `vps.reinstall` script finishes successfully, the Container is started, and the `vps.configure` script is called. At this moment the old private area is mounted to the `/old` directory inside the new one irrespective of the `--skipbackup` option. This is done in order to let you use the necessary files from the old private area in your script, which is to be run inside the running Container. For example, you might want to copy some files from there to regular Container directories.

After the `vps.configure` script finishes, the old private area is either dismounted and deleted or remains mounted depending on whether the `--skipbackup` option was provided.

If you do not want to run these reinstallation scripts and want to stick to the default `prlctl reinstall` behavior, you may do either of the following:

1. Remove the `vps.reinstall` and `vps.configure` scripts from the `/etc/vz/conf` directory, or at least rename them;
2. Modify the last line of the `vps.reinstall` script so that it would read

```
exit 128
```

instead of

```
exit 0
```

The `128` exit code tells the utility not to run the scripts and to reinstall the Container with the default behavior.

Enabling VPN for Containers

Virtual Private Network (VPN) is a technology which allows you to establish a secure network connection even over an insecure public network. Setting up a VPN for a separate Container is possible via the TUN/TAP device. To allow a particular Container to use this device, the following steps are required:

- Make sure the `tun.o` module is already loaded before Parallels Cloud Server is started:

```
# lsmod
```

- Allow the Container to use the TUN/TAP device:

```
# prlctl set 101 --devices c:10:200:rw
```

> **Note:** Currently, the `--devices` option may not be supported by the `prlctl` utility. Use `vzctl` instead.

- Create the corresponding device inside the Container and set the proper permissions:

```
# prlctl exec 101 mkdir -p /dev/net
# prlctl exec 101 mknod /dev/net/tun c 10 200
# prlctl exec 101 chmod 600 /dev/net/tun
```

Configuring the VPN proper is carried out as a common Linux administration task, which is out of the scope of this guide. Some popular Linux software for setting up a VPN over the TUN/TAP driver includes Virtual TUNnel <http://vtun.sourceforge.net/> and OpenVPN <http://openvpn.sourceforge.net/>.

Performing Virtual Machine-Specific Operations

This section focuses on operations specific to virtual machines.

Pausing Virtual Machines

Pausing a running virtual machine releases the resources, such as RAM and CPU, currently used by this virtual machine. The released resources can then be used by the Parallels server or other running virtual machines and Containers.

To pause a virtual machine, you can use the `prlctl pause` command. For example, the following command pauses the `My_VM` virtual machine:

```
# prlctl pause My_VM
Pause the VM...
The VM has been successfully paused.
```

You can check that the virtual machine has been successfully paused by using the `prlctl list -a` command:

```
# prlctl list -a
STATUS    IP_ADDR        NAME
running   10.10.10.101   101
paused    10.10.10.201   My_VM
```

The command output shows that the `My_VM` virtual machine is paused at the moment. To continue running this virtual machine, execute this command:

```
# prlctl start My_VM
Starting the VM...
The VM has been successfully started.
```

Managing Virtual Machine Devices

Parallels Cloud Server allows you to manage the following virtual machine devices:

- hard disk drives
- CD/DVD-ROM drives
- floppy disk drives
- network adapters
- serial and parallels ports
- sound cards
- USB controllers

The main operations you can perform on these devices are:

- adding a new device to the virtual machine
- configuring the device properties
- removing a device from the virtual machine

Adding New Devices

This section provides information on adding new devices to your virtual machines. You can add new virtual devices to your virtual machine using the `prlctl set` command. The options responsible for adding particular devices are listed in the following table:

Option	Description
`hdd`	Adds a new hard disk drive to the virtual machine. You can either connect an existing image to the virtual machine or create a new one. **Note**: SATA disks can be added to running and stopped virtual machines while IDE and SCSI disks—to stopped virtual machines only.
`cdrom`	Adds a new CD/DVD-ROM drive to the virtual machine.
`net`	Adds a new network adapter to the virtual machine.
`fdd`	Adds a new floppy disk drive to the virtual machine.
`serial`	Adds a new serial port to the virtual machine.
`parallel`	Adds a new parallel port to the virtual machine.
`sound`	Adds a new sound device to the virtual machine.
`usb`	Adds a new USB controller to the virtual machine.

For example, you can execute the following command to add a new virtual disk to the `MyVM` virtual machine:

```
# prlctl set MyVM --device-add hdd
Creating hdd1 (+) sata:0 image='/var/parallels/MyVM.pvm/harddisk1.hdd
Create the expanding disk, 65536...
The VM has been successfully configured.
```

This command creates a new virtual disk with the following default parameters:

- name: `hdd1`

- disk type: SATA
- image file name and location: `/var/parallels/MyVM.pvm/harddisk1.hdd`
- disk format: expanding
- disk capacity: 65536 MB

You can redefine some of these parameters by specifying specific options during the command execution. For example, to create an IDE virtual disk that will have the capacity of 84 GB, you can run this command:

```
# prlctl set MyVM --device-add hdd --size 84000 --iface ide
Creating hdd1 (+) ide:1 image='/var/parallels/MyVM.pvm/harddisk1.hdd
Create the expanding disk, 84000Mb...
The VM has been successfully configured.
```

The virtual disk has been added to your virtual machine. However, before starting to use it, you must initialize the disk. Refer to the next subsection for information on how you can do it.

When managing devices, keep in mind the following:

- Detailed information on all options that can be passed to `prlctl set` when creating a new virtual machine device is provided in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.
- You can connect up to 4 IDE devices, 6 SATA devices, and 15 SCSI devices (virtual disks or CD/DVD-ROM drives) to a virtual machine.
- If you want to use an existing image file as a virtual CD/DVD-ROM drive, keep in mind that Parallels Cloud Server supports `.iso`, `.cue`, `.ccd` and `.dmg` (non-compressed and non-encrypted) image files.
- A virtual machine can have only one floppy disk drive.
- A virtual machine can have up to 16 virtual network adapters.
- A virtual machine can have up to four serial ports.
- A virtual machine can have up to three parallel ports.
- Any virtual machine can have only one sound device.
- A virtual machine can have only one USB controller.

Initialize a Newly Added Disk

After you added a new blank virtual hard disk to the virtual machine configuration, it will be invisible to the operating system installed inside the virtual machine until the moment you initialize it.

Initializing the New Virtual Hard Disk in Windows

To initialize a new virtual hard disk in a Windows guest OS, you will need the Disk Management utility available. For example, in Windows 7 and Windows XP you can access this utility by doing the following:

- In Windows 7, click **Start** > **Control Panel** > **System and Security** > **Administrative Tools** >

Computer Management Storage > **Disk Management**.

- In Windows XP, click **Start** > **Control Panel** > **Administrative Tools** > **Computer Management** > **Storage** > **Disk Management**.

When you open the Disk Management utility, it automatically detects that a new hard disk was added to the configuration and launches the **Initialize and Convert Disk** wizard:

1. In the introduction window, click **Next**.
2. In the **Select Disks to Initialize** window, select the newly added disk and click **Next**.
3. In the **Select Disks to Convert** window, select the newly added disk and click **Finish**.

The added disk will appear as a new disk in the Disk Management utility window, but its memory space will be unallocated. To allocate the disk memory, right-click this disk name in the Disk Management utility window and select **New Simple Volume** in Windows Vista or **New Volume** in Windows XP. The **New Simple Volume Wizard**/**New Volume Wizard** window will appear. Follow the steps of the wizard and create a new volume in the newly added disk.

After that your disk will become visible in **Computer**/**My Computer** and you will be able to use it as a data disk inside your virtual machine.

Initializing the New Virtual Hard Disk in Linux

Initializing a new virtual hard disk in a Linux guest OS comprises two steps: (1) allocating the virtual hard disk space and (2) mounting this disk in the guest OS.

To allocate the space, you need to create a new partition on this virtual hard disk using the `fdisk` utility:

Note: To use the `fdisk` utility, you need the `root` privileges.

1. Launch a terminal window.
2. To list the IDE disk devices present in your virtual machine configuration, enter:

```
fdisk /dev/hd*
```

Note: If you added a SCSI disk to the virtual machine configuration, use the `fdisk /dev/sd*` command instead.

3. By default, the second virtual hard disk appears as `/dev/hdc` in your Linux virtual machine. To work with this device, enter:

```
fdisk /dev/hdc
```

Note: If this is a SCSI disk, use the `fdisk /dev/sdc` command instead.

4. To get detailed information about the disk, enter:

```
p
```

5. To create a new partition, enter:

```
n
```

6. To create the primary partition, enter:

```
p
```

7 Specify the partition number. By default, it is `1`.

8 Specify the first cylinder. If you want to create a single partition on this hard disk, use the default value.

9 Specify the last cylinder. If you want to create a single partition on this hard disk, use the default value.

10 To create a partition with the specified settings, enter:

```
w
```

When you allocated the space on the newly added virtual hard disk, you should format it by entering the following command in the terminal:

```
mkfs -t <FileSystem> /dev/hdc1
```

> **Note:** *<FileSystem>* stands for the file system you want to use on this disk. It is recommended to use `ext3` or `ext2`.

When the added virtual hard disk is formatted, you can mount it in the guest OS.

1 To create a mount point for the new virtual hard disk, enter:

```
mkdir /mnt/hdc1
```

> **Note:** You can specify a different mount point.

2 To mount the new virtual hard disk to the specified mount point, enter:

```
mount /dev/hdc1 /mnt/hdc1
```

When you mounted the virtual hard disk, you can use its space in your virtual machine.

Configuring Virtual Devices

In Parallels Cloud Server, you can use the `--device-set` option of the `prlctl set` command to configure the parameters of an existing virtual device. As a rule, the process of configuring the device properties includes two steps:

1 Finding out the name of the device you want to configure.

2 Running the `prlctl set` command to configure the necessary device properties.

Finding Out Device Names

To configure a virtual device, you need to specify its name when running the `prlctl set` command. If you do not know the device name, you can use the `prlctl list` command to learn it. For example, to obtain the list of virtual devices in the `MyVM` virtual machine, run this command:

```
# prlctl list --info MyVM
...
Hardware:
  cpu 2 VT-x accl=high mode=32
  memory 256Mb
  video 46Mb
  fdd0 (+) real='/dev/fd0' state=disconnected
  hdd0 (+) sata:0 image='/var/parallels/MyVM.pvm/harddisk.hdd' 27000Mb
```

```
  hdd1 (+) scsi:0 image='/var/parallels/MyVM.pvm/harddisk1.hdd' 32768Mb
  cdrom0 (+) ide:1 real='Default CD/DVD-ROM'
  parallel0 (+) real='/dev/lp0'
  usb (+)
  net0 (+) type=bridged iface='eth1' mac=001C4201CED0
...
```

All virtual devices currently available to the virtual machine are listed under `Hardware`. In our case the `MyVM` virtual machine has the following devices: 2 CPUs, main memory, video memory, a floppy disk drive, 2 hard disk drives, a CD/DVD-ROM drive, a parallel port, a USB controller, and a network card.

Configuring Virtual Devices

Once you know the virtual device name, you can configure its properties. For example, you can execute the following command to configure the current type of the virtual disk `hdd1` in the `MyVM` virtual machine from SATA to SCSI:

```
# prlctl set MyVM --device-set hdd1 --iface scsi
The VM has been successfully configured.
```

To check that the virtual disk type has been successfully changed, use the `prlctl list --info` command:

```
# prlctl list --info MyVM
...
  hdd0 (+) scsi:1 image='/var/parallels/MyVM.pvm/harddisk.hdd' 85000Mb
...
```

Connecting and Disconnecting Virtual Devices

In Parallels Cloud Server, you can connect or disconnect certain devices when a virtual machine is running. These devices include:

- SATA hard drives
- CD/DVD-ROM drives
- floppy disk drives
- network adapters
- printer ports
- serial ports
- sound devices
- USB devices
- shared folders

Usually, all virtual devices are automatically connected to a virtual machine when you create them. To disconnect a device from the virtual machine, you can use the `prlctl set` command. For example, the following command disconnects the CD/DVD-ROM drive `cdrom0` from the `MyVM` virtual machine:

```
# prlctl set MyVM --device-disconnect cdrom0
```

```
Disconnect device: cdrom0
The VM has been successfully configured.
```

To connect the CD/DVD-ROM drive back, you can run the following command:

```
# prlctl set MyVM --device-connect cdrom0
Connect device: cdrom0
The VM has been successfully configured.
```

Deleting Devices

You can delete a virtual device that you do not need any more in your virtual machine using the `--device-del` option of the `prlctl set` command. The options responsible for removing particular devices are listed in the following table:

Option	Description
`hdd`	Deletes the specified hard disk drive from the virtual machine. **Note**: SATA disks can be removed from running and stopped virtual machines while IDE and SCSI disks—from stopped virtual machines only.
`cdrom`	Deletes the specified CD/DVD-ROM drive from the virtual machine.
`net`	Deletes the specified network adapter from the virtual machine.
`fdd`	Deletes the floppy disk drive from the virtual machine.
`serial`	Deletes the specified serial port from the virtual machine.
`parallel`	Deletes the specified parallel port from the virtual machine.
`sound`	Deletes the sound device from the virtual machine.
`usb`	Deletes the USB controller from the virtual machine.

As a rule deleting a virtual device involves performing two operations:

1. Finding out the name of the device to be deleted.
2. Deleting the device from the virtual machine.

Finding Out the Device Name

To remove a virtual device, you need to specify its name when running the `prlctl set` command. If you do not know the device name, you can use the `prlctl list` command to learn it. For example, to obtain the list of virtual devices in the `MyVM` virtual machine, run this command:

```
# prlctl list --info MyVM
...
Hardware:
  cpu 2 VT-x accl=high mode=32
  memory 256Mb
  video 46Mb
  fdd0 (+) real='/dev/fd0' state=disconnected
  hdd0 (+) ide:0 image='/var/parallels/MyVM.pvm/harddisk.hdd' 27Mb
  hdd1 (+) scsi:0 image='/var/parallels/MyVM.pvm/harddisk1.hdd' 32768Mb
  cdrom0 (+) ide:1 real='Default CD/DVD-ROM'
  parallel0 (+) real='/dev/lp0'
```

```
  usb (+)
  net0 (+) type=bridged iface='eth1' mac=001C4201CED0
...
```

All virtual devices currently available to the virtual machine are listed under `Hardware`. In our case the `MyVM` virtual machine has the following devices: 2 CPUs, main memory, video memory, a floppy disk drive, 2 hard disk drives, a CD/DVD-ROM drive, a parallel port, a USB controller, and a network card.

Deleting a Virtual Device

Once you know the virtual device name, you can remove it from your virtual machine. For example, you can execute the following command to remove the virtual disk `hdd1` from the `MyVM` virtual machine:

```
# prlctl set MyVM --device-del hdd1
Remove the hdd1 device.
The VM has been successfully configured.
```

When deleting virtual machine devices, keep in mind the following:

- If you do not want to permanently delete a virtual device, you can temporarily disconnect if from the virtual machine using the `--disable` option.
- Detailed information on all options that can be used with prlctl set when deleting a device is given in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Making Screenshots

In Parallels Cloud Server, you can use the `prlctl capture` command to capture an image (or screenshot) of your virtual machine screen. You can take screenshots of running virtual machines only. The session below demonstrates how to take a screenshot of the `MyVM` virtual machine screen and save it to the `/usr/screenshots/image1.png` file:

1 Make sure that the virtual machine is running:

```
# prlctl list
STATUS    IP_ADDR         NAME
running   10.10.10.101    101
running    10.10.10.201    MyVM
```

2 Take the virtual machine screenshot:

```
# prlctl capture MyVM --file /usr/screenshots/image1.png
```

3 Check that the `image1.png` file has been successfully created:

```
# ls /usr/screenshots/
image1.png
```

Assigning USB Devices to Virtual Machines

In Parallels Cloud Server, you can assign a USB device to a virtual machine so that the device is automatically connected to the virtual machine when you connect the USB device to the Parallels

server or start the virtual machine. To assign a USB device to a virtual machine, you need to specify two parameters:

- *ID of the USB device*. To get this information, use the `prlsrvctl info` command, for example:

```
# prlsrvctl info
...
Hardware info:
     hdd                                                '/dev/sda'
hdd-part  NTFS                                          '/dev/sda2'
hdd-part  Linux                                         '/dev/sda3'
hdd-part  Linux                                         '/dev/sda5'
hdd-part  Linux swap                                    '/dev/sda6'
 cdrom Optiarc DVD RW AD-7260S                         '/dev/scd0'
   net eth0                                            'eth0'
   usb Broadcom - USB Device 3503                      '2-1.4.3|0a5c|3503|full|KM|Empty'
   usb Broadcom - USB Device 3502                      '2-1.4.2|0a5c|3502|full|KM|Empty'
   usb LITEON Technology - USB Multimedia Keyboard '1-1.6|046d|c312|low|KM|Empty'
 serial  /dev/ttyS0                                      '/dev/ttyS0'
 serial  /dev/ttyS1                                      '/dev/ttyS1'
```

  All USB devices available on the Parallels server are listed in the **Hardware info** section and start with `usb`.

- *ID of the virtual machine*. To get this information, use the `prlctl list --info` command, for example:

```
# prlctl list --info
ID: {d8d516c9-dba3-dc4b-9941-d6fad3767035}
Name: Windows 7
...
```

  The first line in the command output indicates the virtual machine ID; in our case, it is `{d8d516c9-dba3-dc4b-9941-d6fad3767035}`.

Once you know the USB device and virtual machine IDs, you can use the `prlsrvctl usb set` command to assign the USB device to the virtual machine. For example:

```
# prlsrvctl usb set '1-1.6|046d|c312|low|KM|Empty' {d8d516c9-dba3-dc4b-9941-
d6fad3767035}
The server has been successfully configured.
```

This command assigns the USB device `LITEON Technology - USB Multimedia Keyboard` with ID `'1-1.6|046d|c312|low|KM|Empty'` to the virtual machine with ID `{d8d516c9-dba3-dc4b-9941-d6fad3767035}`. When running the command, remember to specify the single quotes and curly brackets with the USB device and virtual machine IDs, respectively.

To check that the USB device has been successfully assigned to the virtual machine, use the `prlsrvctl usb list` command:

```
# prlsrvctl usb list
Broadcom - USB Device 3503                    '2-1.4.3|0a5c|3503|full|KM|Empty'
Broadcom - USB Device 3502                    '2-1.4.2|0a5c|3502|full|KM|Empty'
LITEON Technology - USB Multimedia Keyboard '1-1.6|046d|c312|low|KM|Empty'
       {d8d516c9-dba3-dc4b-9941-d6fad3767035}
```

The command output shows that the USB device with ID `'1-1.6|046d|c312|low|KM|Empty'` is now associated with the virtual machine with ID `{d8d516c9-dba3-dc4b-9941-d6fad3767035}`. This means that the device is automatically connected to the virtual machine every time you start this virtual machine and connect the device to the Parallels server.

To remove the assignment of the USB device with ID `'1-1.6|046d|c312|low|KM|Empty'`, use the `prlsrvctl usb del` command:

```
# prlsrvctl usb del '1-1.6|046d|c312|low|KM|Empty'
The server has been successfully configured.
```

When assigning USB devices to virtual machines, keep in mind the following:

- You cannot migrate a running virtual machine having one or more USB devices assigned.
- After migrating a stopped virtual machine, all its assignments are lost.
- All USB assignments are preserved if you restoring a virtual machine to its original location and are lost otherwise.
- The USB device assignment and a virtual machine is created for the user currently logged in to the system.

Configuring IP Address Ranges for Host-Only Networks

All virtual machines connected to networks of the host-only type receive their IP addresses from the Parallels DHCP server. This DHCP server is set up during the Parallels Cloud Server installation and includes by default IP addresses from 10.37.130.1 to 10.37.130.254. You can redefine the default IP address range for host-only networks and make virtual machines get their IP addresses from different IP address ranges. For example, you can run the following command to set the start and end IP addresses for the `Host-Only` network (this network is automatically created during the Parallels Cloud Server installation) to 10.10.11.1 and 10.10.11.254, respectively:

```
# prlsrvctl net set Host-Only --ip-scope-start 10.10.11.1 --ip-scope-end 10.10.11.254
```

You can also specify a custom IP address range directly when creating a new network of the host-only type. Assuming that you want to create a network with the `Host-Only2` name and define for this network the IP addresses range from 10.10.10.1 to 10.10.10.254, you can execute the following command:

```
# prlsrvctl net add Host-Only2 -t host-only --ip-scope-start 10.10.10.1 --ip-scope-end
10.10.10.254
```

When working with IP address ranges, pay attention to the following:

- The start and end IP addresses of an IP address range must belong to the same subnetwork.
- IP address ranges can be defined for each network of the host-only type separately. For example, you can set the IP address range from 10.10.11.1 to 10.10.11.254 for the `Host-Only` network and from 10.10.10.1 to 10.10.10.254 for the `Host-Only2` network.

Converting Third-Party Virtual Machines and Disks

In Parallels Cloud Server, you can convert third-party virtual machines and their disks to Parallels virtual machines and disks. Currently, you can convert the following third-party virtual machines and disks:

- Microsoft Hyper-V
- Microsoft Virtual PC
- Virtual Box
- VMware

Converting Virtual Machines

Let us assume that you want to convert a VMware virtual machine that runs the CentOS 5 operating system and has the name `centos5`. As the `prlctl convert` command can work only with virtual machines and disks that are available locally, you first need to copy the virtual machine to the Parallels server. Once the virtual machine is on your local server, you can start the conversion. Assuming that you have copied the virtual machine to the `/var/parallels` directory on the Parallels server and the full path to its configuration file is `/var/parallels/centos5/config.xml`, you can run the following command to perform the conversion:

```
# prlctl convert /var/parallels/centos5/config.xml
```

Once the conversion is complete, you can start the virtual machine and manage it in the same way you would manage a native Parallels virtual machine.

Converting Disks

You can also convert third-party virtual disks to Parallels virtual machines and disks using the `prl_convert` utility. Once you run the utility, it checks the disk and, depending on its type, does one of the following:

- If the disk is a system disk—that is, has an operating system installed, `prl_convert` converts it to a Parallels virtual machine. If the utility cannot create a virtual machine for the disk (for example, it fails to detect the operating system on the disk), the disk is converted to a Parallels virtual disk. You can also specify the `--allow-no-os` option to force the conversion, but in this case you may have problems with starting and using the resulting virtual machine.
- If the disk is a data disk, `prl_convert` converts it to a Parallels virtual disk.

When converting third-party virtual disks, you need to specify the full path to the original disk file. For example, to convert the system disk of the `centos5` virtual machine (that is, the disk where the CentOS 5 operating system is installed) that has the full path of `/var/parallels/centos5/centos5.vhd`, you can use this command:

```
# prl_convert /var/parallels/centos5/centos5.vhd
```

This command creates a ready-to-use Parallels virtual machine with the name `centos5`. You can start this virtual machine and manage it in the same way you would manage a native Parallels virtual machine. At the same time, if you convert a third-party virtual data disk, you will need first to add the resulting disk to an existing Parallels virtual machine using the `prlctl set` command.

Notes:

1. When adding a converted virtual disk to an existing Parallels virtual machine or creating a new virtual machine on its basis, make sure that the interface type of the disk is the same as it was in the source virtual machine. For example, if the original disk had the SCSI interface type, ensure that the interface type of the converted disk is also set to SCSI. If you do not configure the disk interface type, it will be set to SATA (this is the default interface type in Parallels virtual machines), which may cause your virtual machine to malfunction.

2. In the current version of Parallels Cloud Server, Hyper-V virtual machines can be converted using the `prl_convert` utility only. That means that you first need to convert all Hyper-V virtual disks and then add them to an existing virtual machine.

CHAPTER 4

Managing Resources

The main goal of resource control in Parallels Cloud Server is to provide Service Level Management or Quality of Service for virtual machines and Containers. Correctly configured resource control settings prevent serious impacts resulting from the resource over-usage (accidental or malicious) of any virtual machine and Container on the other virtual machines and Containers. Using resource control parameters for resources management also allows you to enforce fairness of resource usage among virtual machines and Containers and better service quality for preferred virtual machines and Containers, if necessary.

In This Chapter

What are Resource Control Parameters? 77
Managing CPU Resources..... 77
Managing Disk Quotas 83
Managing Virtual Disks 86
Managing Network Accounting and Bandwidth..... 90
Managing Disk I/O Parameters 96
Managing Memory Parameters for Containers..... 101
Managing Memory Parameters for Virtual Machines 105
Managing Container Resources Configuration..... 110
Managing Virtual Machine Configuration Samples 113
Monitoring Resources..... 114

What are Resource Control Parameters?

The system administrator can control the resources available to a virtual machine and Container through a set of resource management parameters. All these parameters can be set and configured using Parallels command-line utilities.

Managing CPU Resources

In the current version of Parallels Cloud Server, you can manage the following CPU resource parameters for virtual machines and Containers:

- CPU units for virtual machines and Containers (p. 78)
- CPU affinity for virtual machines and Containers (p. 78)

- CPU limits for virtual machines and Containers (p. 79)
- NUMA nodes for Containers (p. 81)
- CPU hotplug for virtual machines (p. 82)

Detailed information on these parameters is given in the following sections.

Configuring CPU Units

CPU units define how much CPU time one virtual machine or Container can receive in comparison with the other virtual machines and Containers on the server if all the CPUs of the server are fully used. For example, if Container 101 and the `MyVM` virtual machine are set to receive 1000 CPU units each and Container 102 is configured to get 2000 CPU units, Container 102 will get twice as much CPU time as Containers 101 or the `MyVM` virtual machine if all the CPUs of the Node are completely loaded.

By default, each virtual machine and Container on the Node gets 1000 CPU units. You can configure the default setting using the `prlctl set` command. For example, you can run the following commands to allocate 2000 CPU units to Container 101 and the `MyVM` virtual machine:

```
# prlctl set 101 --cpuunits 2000
# prlctl set MyVM --cpuunits 2000
```

Configuring CPU Affinity for Virtual Machines and Containers

If your physical server has several CPUs installed, you can bind a virtual machine or Container to specific CPUs so that only these CPUs are used to handle the processes running in the virtual machine or Container. The feature of binding certain processes to certain CPUs is known as *CPU affinity*. Establishing CPU affinity between virtual machines and Containers and physical processors may help you increase your system performance by up to 20%.

By default, any newly created virtual machine and Container can consume the CPU time of all processors installed on the physical server. To bind a virtual machine or Container to specific CPUs, you can use the `--cpumask` option of the `prlctl set` command. Assuming that your physical server has 8 CPUs, you can make the processes in the `MyVM` virtual machine and Container 101 run on CPUs 0, 1, 3, 4, 5, and 6 by running the following commands:

```
# prlctl set MyVM --cpumask 0,1,3,4-6
# prlctl set 101 --cpumask 0,1,3,4-6
```

You can specify the CPU affinity mask—that is, the processors to bind to virtual machines and Containers—as separate CPU index numbers (0,1,3) or as CPU ranges (4-6). If you are setting the CPU affinity mask for a running virtual machine or Container, the changes are applied on the fly.

To undo the changes made to the `MyVM` virtual machine and Container 101 and set their processes to run on all available CPUs on the server, run these commands:

```
# prlctl set MyVM --cpumask all
# prlctl set 101 --cpumask all
```

Configuring CPU Limits for virtual machines and Containers

CPU limit indicates the maximum CPU power a virtual machine or Container may get for its running processes. The Container is not allowed to exceed the specified limit even if the server has enough free CPU power. By default, the CPU limit parameter is disabled for all newly created virtual machines and Containers. This means that any application in any virtual machine or Container can use all the free CPU power of the server.

To set a CPU limit for a virtual machine or Container, you can use one of these options:

- `--cpulimit`
- `--cpus`

Both options are described below in detail.

Using --cpulimit to set CPU limits

As a rule, you set a CPU limit for a virtual machine or Container by using the `--cpulimit` option with the `prlctl set` command. In the following example, Container 101 is set to receive no more than 25% of the server CPU time even if the CPUs on the server are not fully loaded:

```
# prlctl set 101 --cpulimit 25
```

This command sets the CPU limit for Container 101 to 25% of the total CPU power of the server. The total CPU power of a server in per cent is calculated by multiplying the number of CPU cores installed on the server by 100%. So if a server has 2 CPU cores, 2 GHz each, the total CPU power will equal 200% and the limit for Container 101 will be set to 500 MHz.

Now imagine the situation when you migrate Container 101 to another server with 2 CPU cores, 3 GHz each. On this server, Container 101 will be able to get 25% of 6 GHz—that is, 750 MHz. To ensure that Container 101 always has the same CPU limit on all servers, irrespective of their total CPU power, you can set the CPU limits in megahertz (MHz). For example, to make Container 101 consume no more than 500 MHz on any server, run the following command:

```
# prlctl set 101 --cpulimit 500m
```

Note: For more information on setting CPU limits for virtual machines and Containers, see also **CPU Limit Specifics**.

Using --cpus to set CPU limits

Another way of setting a CPU limit for a virtual machine or Container is to use the `--cpus` option with the `prlctl set` command. You may want to use this command if your server has several CPU cores. In this case, you can specify the desired number of CPU cores after `--cpus` and the CPU limit will be set to the sum of CPU powers of the specified cores. For example, if a server has 4 CPU cores, 3 GHz each, you can allow Container 101 to consume no more than 6 GHz of CPU power by running this command:

```
# prlctl set 101 --cpus 2
```

To check that the CPU limit has been successfully set, you can execute the following command:

```
# vzlist -o cpulimitM 101
 CPUL_M
     6000
```

As you can see, the CPU limit for Container 101 is now set to 6000 MHz (or 6 GHz).

Note: To check the CPU limit set for a virtual machine, run the `prlctl list -i` *VM_Name* command and check the value of the `cpulimit` parameter in the command output.

Along with setting the CPU limit for a virtual machine or Container on the server, the `--cpus` option also defines how the information on available CPUs is shown to users. So in the example above, if you log in to Container 101 and run the `cat /proc/cpuinfo` command, you will see that only 2 CPUs are installed in the Container:

```
# prlctl exec 101 cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 15
model           : 4
model name      : Intel(R) Xeon(TM) CPU 3.00GHz
stepping        : 1
cpu MHz         : 2993.581
cache size      : 1024 KB
...

processor       : 1
vendor_id       : GenuineIntel
cpu family      : 15
model           : 4
model name      : Intel(R) Xeon(TM) CPU 3.00GHz
stepping        : 1
cpu MHz         : 2993.581
cache size      : 1024 KB
...
```

Using --cpulimit and --cpus simultaneously

If you use both parameters (`--cpulimit` and `--cpus`) to set the CPU limit for a virtual machine or Container, the smallest limit applies. For example, running the following commands on a server with 4 CPUs, 2 GHZ each, will set the limit for Container 101 to 2 GB:

```
# prlctl set 101 --cpus 2
# prlctl set 101 --cpulimit 2000m
```

CPU Limit Specifics

Internally, Parallels Cloud Server sets the CPU limit for virtual machines and Containers in percent. On multi-core systems, each physical CPU core is considered to have the CPU power of 100%. So if a server has 4 CPU cores, the total CPU power of the server equals 400%.

You can also set a CPU limit in megahertz (MHz). If you specify the limit in MHz, Parallels Cloud Server uses the following formula to convert the CPU power of the server from MHz into percent:

```
CPULIMIT_% = 100% * CPULIMIT_MHz / CPUFREQ
```

where

- `CPULIMIT_%` is the total CPU power of the server in percent.
- `CPULIMIT_MHz` is the total CPU power of the server in megahertz.
- `CPUFREQ` is the CPU frequency of one core on the server.

When setting CPU limits, notice the following:

- Make sure that the CPU limit you plan to set for a virtual machine or Container does not exceed the total CPU power of the server. So if a server has 4 CPUs, 1000 MHz each, do not set the CPU limit to more than 4000 MHz.
- The processes running in a virtual machine or Container are scheduled for execution on all server CPUs in equal shares. For example, if a server has 4 CPUs, 1000 MHz each, and you set the CPU limit for a virtual machine or Container to 2000 MHz, the virtual machine or Container will consume 500 MHz from each CPU.
- All running virtual machines and Containers on a server cannot simultaneously consume more CPU power than is physically available on the node. In other words, if the total CPU power of the server is 4000 MHz, the running virtual machines and Containers on this server will not be able to consume more than 4000 MHz, irrespective of their CPU limits. It is, however, perfectly normal that the overall CPU limit of all virtual machines and Containers exceeds the Node total CPU power because most of the time virtual machines and Containers consume only part of the CPU power assigned to them.

Binding CPUs to NUMA Nodes

On systems with a NUMA (Non-Uniform Memory Access) architecture, you can configure Containers to use CPUs from specific NUMA nodes only. Let us assume the following:

- Your physical server has 8 CPUs installed.
- The CPUs are divided into 2 NUMA nodes: NUMA node 0 and NUMA node 1. Each NUMA node has 4 CPUs.
- You want the processes in Container 101 to be executed on the processors from NUMA node 1.

To set Container 101 to use the processors from NUMA node 1, run the following command:

```
# prlctl set 101 --nodemask 1
```

To check that Container 101 is now bound to NUMA node 1, use this command:

```
# vzlist 101 -o nodemask
         NODEMASK
                1
```

To unbind Container 101 from NUMA node 1, execute this command:

```
# prlctl set 101 --nodemask all
```

Now Container 101 should be able to use all CPUs on the server again.

Note: For more information on NUMA, visit http://lse.sourceforge.net/numa.

Enabling CPU Hotplug for Virtual Machines

If a guest operating system supports the CPU hotplug functionality, you can enable this functionality for the virtual machine. Once the CPU hotplug functionality is turned on, you can increase the number of CPUs available to your virtual machines even if they are running.

Note: Decreasing the number of CPUs available to a running virtual machine is not supported in the current version of Parallels Cloud Server.

Currently, the following systems come with the CPU hotplug support:

Linux (both x86 and x64 versions)

- Linux operating systems based on the RHEL 5 kernel and higher (Red Hat Linux Enterprise 5, CentOS 5, and so on)

Windows

- x64 version of Windows Server 2008 (Standard Edition)
- x64 version of Windows Server 2008 (Enterprise Edition)
- x64 version of Windows Server 2008 (Datacenter Edition)
- x64 version of Windows Server 2008 R2 (Datacenter Edition)

By default, the CPU hotplug support is disabled for all newly created virtual machines. To enable this functionality, you can use the `--cpu-hotplug` option of the `prlctl set` command. For example, to enable the CPU hotplug support in the `MyVM` virtual machine that runs one of the supported operating systems, stop the `MyVM` virtual machine and run this command:

```
# prlctl set MyVM --cpu-hotplug on
set cpu hotplug: 1
The VM has been successfully configured.
```

Once the functionality is enabled, you can increase the number of CPUs in the `MyVM` virtual machine even it is running. Assuming that your physical server has 4 CPUs installed and the processes in the `MyVM` virtual machine are set to be executed on two CPUs, you can run the following command to assign 3 CPUs to the virtual machine:

```
# prlctl set MyVM --cpus 3
set cpus(4): 3
The VM has been successfully configured.
```

To disable the CPU hotplug support in the `MyVM` virtual machine, use this command:

```
# prlctl set MyVM --cpu-hotplug off
set cpu hotplug: 0
The VM has been successfully configured.
```

The changes will come into effect on the next virtual machine start.

Managing Disk Quotas

This section explains the basics of disk quotas, describes disk quota parameters as well as the following operations:

- enabling and disabling per-Container quotas (p. 84)
- setting Per-Container quotas (p. 85)
- enabling and disabling per-user and per-group quotas in Containers (p. 85)
- setting per-user and per-group quotas from inside Containers (p. 85)

What are Disk Quotas?

In the current version of Parallels Cloud Server, system administrators can limit the amount of disk space Containers can use. Such quotas are known as per-Container or first-level quotas. In addition, administrators can enable or disable per-user and per-group quotas that are known as second-level quotas and allow you to limit disk space that individual users and groups in a Container can use.

By default, first-level quotas on your server are enabled (as defined in the `/etc/vz/vz.conf` configuration file), whereas second-level quotas must be turned on for each Container separately (in the corresponding Container configuration files). It is impossible to turn on second-level disk quotas for a Container if first-level disk quotas are off.

Disk Quota Parameters

The table below summarizes the disk quota parameters that you can control. The **File** column indicates that the parameter is defined in a Container configuration file (V) or in the global configuration file but can be overridden in a Container configuration file (GV).

Parameter	Description	File
`DISK_QUOTA`	Enables or disables per-Container quotas for all or particular Containers.	GV
`DISKSPACE`	The total disk space a Container may consume, in kilobytes.	V

QUOTAUGIDLIMIT	Configures per-user and per-group quotas: • For Containers with the Container-in-an-image-file layout), it enables (if set to a value other than 0) and disables (if set to 0) quotas. • For VZFS Containers, it sets the maximum aggregate number of user IDs and group IDs for which disk quota will be accounted. If set to 0, the UID and GID quota are disabled. **Note:** This guide describes how to manage per-user and per-group quotas for Containers with the Container-in-an-image-file layout. For information on managing these quotas for VZFS Containers, consult the documentation for Parallels Server Bare Metal 5.0.	V

Managing Per-Container Disk Quotas

This section explains how to manage per-Container disk quotas.

Enabling and Disabling Per-Container Quotas

Per-Container disk quotas can be enabled or disabled with the DISK_QUOTA parameter in the global configuration file (/etc/vz/vz.conf). The same parameter in a Container configuration file (/etc/vz/conf/<*CT_ID*>.conf) overrides the one in the global configuration file. To enable quota support for some Containers and disable it for other, it is recommended to set DISK_QUOTA to yes in the global configuration file and then set it to no in the corresponding Container configuration files.

In the example below, per-Container quotas are enabled globally but disabled for Container 101:

1 Check that quota is enabled.

```
# grep DISK_QUOTA /etc/vz/vz.conf
DISK_QUOTA=yes
```

2 Check available space on the /vz partition.

```
# df /vz
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/sda2              8957295   1421982   7023242  17% /vz
```

3 Set DISK_QUOTA to no in the Container 101 configuration file.

```
# vi /etc/vz/conf/101.conf
```

4 Check that quotas are disabled for Container 101.

```
# grep DISK_QUOTA /etc/vz/conf/101.conf
DISK_QUOTA=no
# prlctl start 101
# prlctl exec 101 df
Filesystem           1k-blocks      Used  Available  Use%  Mounted on
/dev/ploop1p1          8282373    747060    7023242   10%  /
```

As shown in the example above, a Container with quotas disabled has only one disk space limit: the available space on the partition where the Container private area is.

Setting Quota Parameters

To set disk quotas for Containers, you use the `prlctl set` command. In the example below, the disk space available to Container 101 is set to 20 GB:

```
# prlctl set 101 --diskspace 20971520
```

You can check the result as follows:

```
# prlctl exec 101 df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/ploop1p1         20642152    737692  18855888   4% /
none                    131072         4    131068   1% /dev
```

You can change the per-Container disk quota parameters for running Containers. The changes take effect immediately.

Managing Per-User and Per-Group Disk Quotas

This section explains how to manage per-user and per-group disk quotas for Containers.

Enabling and Disabling Quotas

You can enable or disable per-user and per-group disk quotas by using the `prlctl set --quotaugidlimit` command and then restarting the Container.

To enable quotas, use:

```
# prlctl set 100 --quotaugidlimit 1
```

To disable quotas, use:

```
# prlctl set 100 --quotaugidlimit 0
```

Note: This command changes the `QUOTAUGIDLIMIT` parameter in the Container configuration file (`/etc/vz/conf/<CT_ID>.conf`).

Managing Quota Parameters

Parallels Cloud Server provides the standard Linux `quota` package for working inside Containers:

```
# prlctl exec 101 rpm -q quota
quota-3.17-16.el6.x86_64
```

This command shows that the `quota` package installed in the Container is built and shipped by Parallels. Use the utilities from this package (as is prescribed in your Linux manual) to set second-level quotas for the given Container. For example:

```
# prlctl enter 101
CT-101-bash-4.1# edquota root
Disk quotas for user root (uid 0):
  Filesystem      blocks        soft      hard     inodes     soft    hard
  /dev/ploop1p1    38216       50000     60000    45454      70000   70000
CT-101-bash-4.1# repquota -a
*** Report for user quotas on device /dev/ploop1p1
```

```
Block grace time: 00:00; Inode grace time: 00:00
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      --   38218   50000   60000          45453 70000 70000
[the rest of repquota output is skipped]

CT-101-bash-4.1# dd if=/dev/zero of=test
dd: writing to `test': Disk quota exceeded
23473+0 records in
23472+0 records out
CT-101-bash-4.1# repquota -a
*** Report for user quotas on device /dev/ploop1p1
Block grace time: 00:00; Inode grace time: 00:00
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      +-   50001   50000   60000   none   45454 70000 70000
[the rest of repquota output is skipped]
```

The above example shows the session when the `root` user has the disk space quota set to the hard limit of 60,000 1KB blocks and to the soft limit of 50,000 1-KB blocks; both hard and soft limits for the number of inodes are set to 70,000.

It is also possible to set the grace period separately for block limits and inodes limits with the help of the `/usr/sbin/setquota` command. For more information on using the utilities from the `quota` package, consult the system administration guide shipped with your Linux distribution or online manual pages included in the package.

Managing Virtual Disks

In Parallels Cloud Server, you can manage virtual disks as follows:

- increase the capacity of your virtual disks,
- reduce the capacity of your virtual disks,
- compact virtual disks (reduce the size they occupy on the physical hard drive),
- change the type of your virtual disks (currently, virtual machines only).

All these operations are described in the following subsections in detail.

Changing Virtual Machine Disk Type

A virtual disk can be one of the two types:

- `plain`. A plain virtual hard disk has a fixed size from the moment of its creation.
- `expanding`. An expanding virtual hard disk is small initially. Its size grows as you add applications and data to it.

A new virtual machine is created with an expanding virtual disk. However, you can change the type of the virtual disk using either the `prlctl` or `prl_disk_tool` utility. Let us assume that the current type of the `hdd0` virtual disk in the `MyVM` virtual machine is expanding and you want to change it to plain. To do this, you can execute one of the following commands:

```
# prlctl set MyVM --device-set hdd0 --type plain
```

or

```
# prl_disk_tool convert --hdd /var/parallels/MyVM/harddisk.hdd --plain
```

The main difference between these two commands is that `prlctl` requires for its execution the disk name as it is shown by the `prlctl list --info` command (`hdd0`) while `prl_disk_tool` needs the full path to the virtual disk drive (`/var/parallels/MyVM/harddisk.hdd`).

To change the disk type back to expanding, run one of the following commands:

```
# prlctl set MyVM --device-set hdd0 --type expand
```

or

```
# prl_disk_tool convert --hdd /var/parallels/MyVM/harddisk.hdd --expanding
```

Increasing Disk Capacity

If you find that the capacity of the virtual hard disk of your virtual machine or Container does not fit your needs anymore, you can increase it using the `prl_disk_tool` utility. For example, you can execute the following command to set the capacity for the `MyVM-0.hdd` disk to 80 GB:

```
# prl_disk_tool resize --size 80000 --hdd /vz/MyVM.pvm/MyVM-0.hdd/
```

This command adds additional disk space as unallocated space. You can use standard means (e.g., the Disk Management tool in Windows-based virtual machines) to allocate this space by creating a new partition or expanding an existing one.

At the same time, you can use the `--resize_partition` option to automatically add additional space to the last partition on the virtual disk:

```
# prl_disk_tool resize --size 80000 --hdd /vz/MyVM.pvm/MyVM-0.hdd/ --resize_partition
```

When increasing the disk capacity, keep in mind the following:

- You cannot increase the capacity of a virtual disk if the virtual machine or Container using this disk is running.
- The virtual machine or Container using the virtual disk you want to configure must not have any snapshots. Otherwise, the operation will fail:

```
# prl_disk_tool resize --size 68000 --hdd /vz/MyVM.pvm/MyVM-0.hdd/
This disk has one or more snapshots and cannot be resized.
You need to delete snapshots using the prlctl tool before resizing the disk.
```

  In this case, you should delete all existing snapshots and run the command again. To learn how to delete snapshots of a virtual machine or Container, refer to **Deleting Snapshots** (p. 49).

- The capacity of an expanding virtual disk shown from inside the virtual machine or Container

and the size the virtual disk occupies on the server's physical disk may differ.

Reducing Disk Capacity

Parallels Cloud Server provides a possibility to reduce the size of an expanding virtual disk by setting the limit the disk cannot exceed. In general, the process of reducing a virtual disk includes these steps:

1. Finding out the minimum capacity to which the disk can be reduced.
2. Running the `prl_dsk_tool resize` command to reduce the disk.

Checking the minimum disk capacity

Before reducing a virtual disk, you may wish to see the minimum capacity to which it can be reduced. To do this, use the `prl_disk_tool resize --info` command. For example, you can run the following command to get detailed information on the `MyVM-0.hdd` disk:

```
# prl_disk_tool resize --info --hdd /vz/MyVM.pvm/MyVM-0.hdd
Disk information:
        Size:                                           65537M
        Minimum:                                         2338M
        Minimum without resizing the last partition:    65523M
```

The information on the virtual disk is presented in the form of the following table:

Column	Description
`Size`	The virtual disk disk capacity, in megabytes, as it is seen from inside the virtual machine or Container.
`Minimum`	The virtual disk capacity, in megabytes, after resizing the disk using the `prl_disk_tool` utility with the `--resize_partition` option.
`Minimum without resizing the last partition`	The virtual disk capacity, in megabytes, after resizing the disk using the `prl_disk_tool` utility without the `--resize_partition` option.

Reducing disk size

Once you know the minimum capacity of the virtual disk, you can reduce it. For example, to reduce the `MyVM-0.hdd` disk to 30 GB, you can execute the following command:

```
#  prl_disk_tool resize --size 30G --hdd /vz/MyVM.pvm/MyVM-0.hdd --resize_partition
```

When reducing the disk capacity, keep in mind the following:

- You cannot reduce the capacity of a virtual disk if the virtual machine or Container using this disk is running.
- The virtual machine or Container using the virtual disk you want to configure must not have any snapshots. Otherwise, you will be informed of this fact:

```
# prl_disk_tool resize --size 68000 --hdd /vz/MyVM.pvm/MyVM-0.hdd/
This disk has one or more snapshots and cannot be resized.
You need to delete snapshots using the prlctl tool before resizing the disk.
```

In this case, you should delete all existing snapshots and run the command again. To learn how

to delete snapshots of a virtual machine or Container, refer to **Deleting Snapshots** (p. 49).

- The capacity of an expanding virtual disk shown from inside the virtual machine or Container and the size the virtual disk occupies on the server's physical disk may differ.

Compacting Disks

In Parallels Cloud Server, you can decrease the space your virtual machines and Containers occupy on the Parallels server's disk drive by compacting their virtual disks. Compacting virtual disks allows you to save your server's disk space and host more virtual machines and Containers on the server.

Note: Plain disks cannot be compacted.

To compact a virtual disk, you can use the `prl_disk_tool compact` command. For example, to compact the `MyVM-0.hdd` disk, you can run this command:

```
# prl_disk_tool compact --hdd /vz/MyVM.pvm/MyVM-0.hdd/
```

To check the space that was freed by compacting the virtual disk, you can use standard Linux utilities (for example, the `df` utility).

Managing Virtual Machine Disk Interfaces

By default, any virtual machine is created with a SATA (Serial Advanced Technology Attachment) virtual hard disk. If necessary, you can change the interface type of a disk from SATA to SCSI (Small Computer System Interface) or IDE (Integrated Drive Electronics). For example, to change the interface type of the default disk (`hdd0`) in the `MyVM` virtual machine from SATA to SCSI, you can run the following command:

```
# prlctl set MyVM --device-set hdd0 --iface scsi
The VM has been successfully configured
```

To check that the interface type has been successfully changed, use this command:

```
# prlctl list -i MyVM | grep hdd0
Boot order: hdd0 cdrom0 fdd0 net0
  hdd0 (+) scsi:0 image='/var/parallels/VM_SCSI.pvm/harddisk.hdd' 65536Mb
```

The command output shows that now the interface type of the `hdd0` disk is SCSI.

You can create additional disks for the `MyVM` virtual machine. For example, to add a new disk of the SCSI type to the virtual machine, execute the following command:

```
# prlctl set MyVM --device-add hdd --iface scsi
Creating hdd1 (+) scsi:1 image='/var/parallels/MyVM.pvm/harddisk1.hdd' 65536Mb
Create the expanding image file, 65536Mb...
The VM has been successfully configured.
```

You can also create an IDE disk. To do this, specify `--iface ide` instead of `--iface scsi` in the command above. If you omit the `--iface` option, a SATA disk is created by default.

The maximum number of devices (both virtual hard disks and CD/DVD-ROM drives) you can add to a virtual machine is given below:

- 4 IDE devices
- 6 SATA devices
- 15 SCSI devices

At any time, you can remove the `hdd1` disk from the `MyVM` virtual machine:

```
# prlctl set MyVM --device-del hdd1
Remove the hdd1 device.
The VM has been successfully configured.
```

Notes:

1. Virtual SATA disks can be added to or removed from both running and stopped virtual machines while operations with IDE and SCSI disks can be performed on stopped virtual machines only.

2. You need to initialize a newly added disk before you can start using it. To initialize the disk, use standard means provided by your guest operating system.

3. For more information on the `prlctl` utility and its options, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Managing Network Accounting and Bandwidth

This section explains how to perform the following tasks in Parallels Cloud Server:

- configuring network classes
- viewing network traffic statistics
- turning on and off network bandwidth management
- configuring bandwidth limits

Network Traffic Parameters

The table below summarizes the network traffic parameters that you can control in Parallels Cloud Server.

Parameter	Description
`traffic_shaping`	If set to `yes`, traffic limitations for outgoing traffic are set for virtual machines and Containers. The default is `no`.
`bandwidth`	This parameter lists all network adapters installed on the server and their bandwidth.
`totalrate`	This parameter defines the bandwidth to allocate for each network class. It is active if traffic shaping is turned on.

`rate`	If traffic shaping is turned on, this parameter specifies the bandwidth guarantee for virtual machines and Containers.
`ratebound`	If this parameter is set to `yes`, the bandwidth guarantee (the global rate parameter) is also the limit for the virtual machine or Container, and the virtual machine or Container cannot borrow the bandwidth from the `totalrate` bandwidth pool.

Configuring Network Classes

Parallels Cloud Server allows you to track the inbound and outbound network traffic as well as to shape the outgoing traffic for a virtual machine and Container. To provide the ability to distinguish between domestic and international traffic, a concept of network classes is introduced. It is important to fully understand this notion, because network classes IDs are used in the values of some network traffic parameters. A network class is a range of IP addresses for which Parallels Cloud Server counts and shapes the traffic.

Classes are specified in the `/etc/vz/conf/networks_classes` file. The file is in the ASCII format, and all empty lines and lines starting with the `#` sign are ignored. Other lines have the following format:

```
<class_id> <IP_address>/<prefix_length>
```

where *<class_id>* defines the network class ID, and the *<IP_address>*/*<prefix_length>* pair defines the range of IP addresses for this class. There may be several lines for each class.

Classes 0 and 1 have special meanings:

- Class 0 defines the IP address range for which no accounting is performed. Usually, it corresponds to the server subnet (the server itself and its virtual machines and Containers). Setting up class 0 is not required; however, its correct setup improves performance.
- Class 1 is defined by Parallels Cloud Server to match any IP address. It must be always present in the network classes definition file. Therefore, it is suggested not to change the default line in the `networks_classes` file.

```
1 0.0.0.0/0
```

  If your virtual machines and Containers are using IPv6 addresses, you can also add the following line to this file:

```
1 ::/0
```

Other classes should be defined after class 1. They represent exceptions from the "matching-everything" rule of class 1. The example below illustrates a possible configuration of the network classes definition file containing rules for both IPv4 and IPv6 addresses:

```
# server networks
0 192.168.0.0/16
0 fe80::/64

# any IP address (all traffic)
1 0.0.0.0/0
1 ::/0

# class 2 - addresses for the "foreign" traffic
```

```
2 10.0.0.0/8
2 2001:db88::/64

# inside "foreign" network there
# is a hole belonging to "local" traffic
1 10.10.16.0/24
1 2001:db88:3333::/64
```

In this example, IPv4 addresses in the range of `192.168.0.0` to `192.168.255.255` and IPv6 addresses in the range of `fe80::` to `fe80::ffff:ffff:ffff:ffff` are treated as class 0 addresses and no accounting is done for the traffic from virtual machines and Containers destined to these addresses.

Class 2 matches the following IP addresses:

- IPv4 addresses from `10.0.0.0` to `10.255.255.255` with the exception of addresses in the sub-range of `10.10.16.0` to `10.10.16.255`, which are treated as class 1.
- IPv6 addresses from `2001:db88::` to `2001:db88::ffff:ffff:ffff:ffff` with the exception of addresses in the sub-range of `2001:db88:3333::` to `2001:db88:3333::ffff:ffff:ffff:ffff`, which are also treated as class 1.

All other IP addresses (both IPv4 and IPv6) belong to class 1.

Note: After editing the `/etc/vz/conf/networks_classes` file, execute either the `/etc/init.d/vz accrestart` or `service vz accrestart` command for the changes made to the file to take effect.

Viewing Network Traffic Statistics

In Parallels Cloud Server, you can view the current network traffic statistics for virtual machines and Containers using the `pnetstat` utility. For example:

```
# pnetstat
UUID           Net.Class   Input(bytes)  Input(pkts)   Output(bytes)   Output(pkts)
47406484...    0           0             0             0               0
47406484...    1           5867489       37155         58033           1010
49b66605...    0           0             0             0               0
49b66605...    1           7357952       41424         1023555         1023
101            0           0             0             0               0
101            1           58140960      66635         1025931         19408
102            0           0             0             0               0
102            1           0             0             0               0
```

By default, `pnetstat` shows network statistics for both virtual machines and Containers. In the example above, the statistics are displayed for Containers with IDs 101 and 102 and two virtual machines with UUIDs 47406484... and 49b66605.... Keep in mind that `pnetstat` displays statistics only about virtual machines and Containers which were started at least once.

The `pnetstat` utility displays the following information:

Column	Description
UUID	UUID assigned to virtual machine or Container.

Net.Class	ID of the network class for which network statistics is calculated.
Input(bytes)	Amount of incoming traffic, in bytes.
Input(pkts)	Amount of incoming traffic, in packets.
Output(bytes)	Amount of outgoing traffic, in bytes.
Output(pkts)	Amount of outgoing traffic, in packets.

For example, from the command output above, you can see that around 58 MB of data were uploaded to Container 101, (2) about 10 MB were downloaded from it, and all the traffic was exchanged with servers from class 1 networks.

If necessary, you can view network traffic statistics separately for virtual machine or Container by passing the `-t` option to `pnetstat`:

- For Containers only:

```
# pnetstat -t ct
UUID          Net.Class  Input(bytes)  Input(pkts)  Output(bytes)  Output(pkts)
101           0          0             0            0              0
101           1          58140960      66635        1025931        19408
102           0          0             0            0              0
102           1          0             0            0              0
```

- For virtual machines only:

```
# pnetstat -t vm
UUID          Net.Class  Input(bytes)  Input(pkts)  Output(bytes)  Output(pkts)
47406484...   0          0             0            0              0
47406484...   1          5867489       37155        58033          1010
49b66605...   0          0             0            0              0
49b66605...   1          7357952       41424        1023555        1023
```

You can also view network statistics for a particular virtual machine or Container by specifying its ID after the `-v` option, for example:

```
# pnetstat -v 101
UUID          Net.Class  Input(bytes)  Input(pkts)  Output(bytes)  Output(pkts)
101           0          0             0            0              0
101           1          58140960      66635        1025931        19408
```

This command displays statistics only for Container 101.

Turning On and Off Network Bandwidth Management

Traffic shaping (also known as network bandwidth management) allows you to control what network bandwidth a virtual machine or Container receives for outgoing traffic. Traffic shaping is off by default in Parallels Cloud Server and is controlled by the `TRAFFIC_SHAPING` parameter in the `/etc/vz/vz.conf` global configuration file.

Note: Incoming traffic cannot be controlled for virtual machines and Containers in Parallels Cloud Server.

To turn traffic shaping on, you need to complete the following steps:

1 Set the value of `TRAFFIC_SHAPING` to `yes` in the global configuration file.

2 Correctly set up the BANDWIDTH and TOTALRATE parameters values.

3 Start traffic shaping with the /etc/init.d/vz shaperon command.

The BANDWIDTH variable is used for specifying the network rate, in kilobits per second, of available network adapters. By default, it is set to eth0:100000, which corresponds to a 100Mb/s Fast Ethernet card. If your server has more network adapters installed, update this parameter by listing all the adapters participating in shaping. For example, if you have two Fast Ethernet cards, set the parameter to eth0:100000 eth1:100000.

The TOTALRATE variable specifies the size of the so-called bandwidth pool for each network class being shaped. The bandwidth from the pool can be borrowed by virtual machines and Containers when they need more bandwidth for communicating with hosts from the corresponding network class. It is used to limit the total available outgoing traffic virtual machines and Containers can consume. The format of this variable is *<NIC>*:*<network_class>*:*<bandwidth_in_Kbits_per_second>* and defines the pool size per network class for a given network adapter. Multiple entries for different network classes and adapters can be separated by spaces. The default value for TOTALRATE is eth0:1:4000, which corresponds to the pool size of 4Mb/s for Network Class 1 on the first Ethernet adapter.

In the /etc/vz/vz.conf configuration file, you can also define the RATE variable whose value amounts to the number of kilobits per second any virtual machine and Container is guaranteed to receive for outgoing traffic with a network class on an Ethernet device. The default value of this parameter is eth0:1:8, which means that any virtual machine and Container is guaranteed to receive the bandwidth of at least 8 Kbps for sending data to Class 1 hosts on the first Ethernet device. This bandwidth is not the limit for a virtual machine and Container (unless the RATEBOUND parameter is enabled for the virtual machine or Container); the virtual machine or Container can take the needed bandwidth from the TOTALRATE bandwidth pool if it is not used by other virtual machines and Containers.

After setting up the above parameters, start bandwidth management as follows:

```
# /etc/init.d/vz shaperon
Starting shaping: Ok
Set shaping on running Container :
vz WARNING: Can't get tc class for Container(101).
vz WARNING: Can't access file /var/run/vz_tc_classes. \
Creating new one.
vz WARNING: Can't get tc class for Container(1).
```

Now you have activated the network bandwidth limits. To turn traffic shaping off temporarily, use the /etc/init.d/vz shaperoff command. If you want to disable bandwidth management permanently, set the TRAFFIC_SHAPING variable to no in the /etc/vz/vz.conf configuration file.

Configuring Network Bandwidth Management

The network bandwidth for outgoing traffic a virtual machine and Container receives is controlled by two parameters: RATE and RATEBOUND.

Note: Incoming traffic cannot be controlled in the current version of Parallels Cloud Server.

The RATE parameter specifies the guaranteed outgoing traffic rate that a virtual machine or Container receives. This rate can be specified differently for different network classes. Bandwidth values are specified in kilobits per second (Kbps). It is recommended to increase this value in 8 Kbps increments and set it to at least 8 Kbps. The example below demonstrates how to set the RATE parameter for the MyVM virtual machine and Container 101 to 16 Kbps for network class 1 on the eth0 network adapter:

```
# prlctl set MyVM --rate 1:16
# prlctl set 101 --rate eth0:1:16
```

Note: For Containers, you can also configure the RATE parameter for different network adapters. For virtual machines, you can set this parameter for the default network adapter (usually eth0) only. The rates for all other network adapters can be configured in the /etc/vz/vz.conf global configuration file.

The RATEBOUND parameter specifies whether the network bandwidth available to virtual machine or Container for outgoing traffic is limited by the bandwidth specified in the RATE variable. By default, this parameter is turned off for all newly created virtual machines and Containers. That means that virtual machines and Containers are allowed to take free bandwidth from the TOTALRATE pool. You can turn the RATEBOUND parameter on by using the --ratebound option of the prlctl set command, for example:

```
# prlctl set MyVM --ratebound on
# prlctl set 101 --ratebound on
```

The actual network bandwidth available to virtual machines and Containers depends on the number of virtual machines and Containers and the total sum of the RATE values, and normally does not coincide with the bandwidth specified in their own RATE parameters. If the RATEBOUND parameter is turned on, the virtual machine or Container bandwidth is limited by the value of the RATE parameter.

If the the RATE and RATEBOUND parameters are not set for individual virtual machines and Containers, the values from the /etc/vz/vz.conf configuration file are taken. By default, Parallels Cloud Server does not set RATEBOUND, which corresponds to no, and RATE is set to eth0:1:8.

The network bandwidth management in Parallels Cloud Server works in the following way. The bandwidth pool for a given network class (configurable through the TOTALRATE variable in the global configuration file) is divided among the virtual machines and Containers transmitting data proportionally to their RATE settings. If the total value of the RATE variables of all virtual machines and Containers transmitting data does not exceed the TOTALRATE value, each virtual machine or Container gets the bandwidth equal or greater than its RATE value (unless the RATEBOUND variable is enabled for this virtual machine or Container). If the total value of the RATE variables of all virtual machines and Containers transmitting data exceeds the TOTALRATE value, each virtual machine or Container may get less than its RATE value.

The example below illustrates the scenario when Containers 101 and 102 have RATEBOUND set to no, and the MyVM virtual machine has RATEBOUND set to yes. With the default TOTALRATE of 4096 Kbps and RATE of 8 Kbps, the bandwidth pool will be distributed according to the following table:

Container 101	Container 102	MyVM	Consumed Bandwidth
transmits	idle	idle	Container 101: 4096 Kbps
idle	idle	transmits	MyVM: 8 Kbps
transmits	transmits	idle	Container 101: 2048 Kbps Container 102: 2048 Kbps
transmits	idle	transmits	Container 101: 4032 Kbps MyVM: 8 Kbps
transmits	transmits	transmits	Container 101: 2016 Kbps Container 102: 2016 Kbps Container 103: 8 Kbps

Once you configure the bandwidth settings, activate your changes by running the following command:

```
# /etc/init.d/vz shaperrestart
Stopping shaping: Ok
Starting shaping: Ok
Set shaping on running Container: Ok
```

This command clears off all existing shaping settings and sets them again using the configuration files of running virtual machines and Containers.

Managing Disk I/O Parameters

This section explains how to manage disk input and output (I/O) parameters in Parallels Cloud Server systems.

Configuring Priority Levels for Virtual Machines and Containers

In Parallels Cloud Server, you can configure the disk I/O (input/output) priority level of virtual machines and Containers. The higher the I/O priority level, the more time the virtual machine or Container will get for its disk I/O activities as compared to the other virtual machines and Containers on the server. By default, any virtual machine and Container on the server has the I/O priority level set to 4. However, you can change the current I/O priority level in the range from 0 to 7 using the --ioprio option of the prlctl set command. For example, you can issue the following command to set the I/O priority of Container 101 and the MyVM virtual machine to 6:

```
# prlctl set 101 --ioprio 6
# prlctl set MyVM --ioprio 6
```

To check the I/O priority level currently applied to Container 101 and the MyVM virtual machine, you can execute the following commands:

- For Container 101:

```
# grep IOPRIO /etc/vz/conf/101.conf
IOPRIO="6"
```

- For the `MyVM` virtual machine:

```
# prlctl list --info | grep ioprio
cpu 2 VT-x accl=high mode=32 cpuunits=1000 ioprio=6 iolimit=0
```

Configuring Disk I/O Bandwidth

In Parallels Cloud Server, you can configure the bandwidth virtual machines and Containers are allowed to use for their disk input and output (I/O) operations. Limiting the disk I/O bandwidth can help you prevent the situations when high disk activities in one virtual machine or Container (generated, for example, by transferring huge amounts of data to/from the virtual machine or Container) can slow down the performance of other virtual machines and Containers on the Parallels server.

By default, the I/O bandwidth limit for all newly created virtual machines and Containers is set to 0, which means that no limits are applied to any virtual machines and Containers. To limit the disk I/O bandwidth for a virtual machine or Container, you can use the `--iolimit` option of the `prlctl set` command. For example, the following command sets the I/O bandwidth limit for the `MyVM` virtual machine to 10 megabytes per second (MB/s):

```
# prlctl set MyVM --iolimit 10
Set up iolimit: 10485760
The VM has been successfully configured.
```

To set the limit for a Container, just specify its ID instead of the virtual machine name, for example:

```
# prlctl set 101 --iolimit 10
Set up iolimit: 10485760
Saved parameters for Container 101
```

By default, the limit is set in megabytes per second. However, you can use the following suffixes to use other measurement units:

- `G`: sets the limit in gigabytes per second (1G).
- `K`: sets the limit in kilobytes per second (10K).
- `B`: sets the limit in bytes per second (10B).

Note: In the current version of Parallels Cloud Server, the maximum I/O bandwidth limit you can set for a virtual machine and Container is 2 GB per second.

To check that the I/O speed limit has been successfully applied to the `MyVM` virtual machine and Container 101, use the `prlctl list` command:

```
# prlctl list MyVM -o iolimit
110485760
# prlctl list 101 -o iolimit
IOLIMIT
10485760
```

At any time, you can remove the I/O bandwidth limit set for the `MyVM` virtual machine and Container 101 by running these commands:

```
# prlctl set MyVM --iolimit 0
Set up iolimit: 0
The VM has been successfully configured.
# prlctl set 101 --iolimit 0
Set up iolimit: 0
Saved parameters for Container 101
```

Configuring the Number of I/O Operations Per Second

In Parallels Cloud Server, you can limit the maximum number of disk input and output operations per second virtual machines and Containers are allowed to perform (known as the IOPS limit). You may consider setting the IOPS limit for virtual machines and Containers with high disk activities to ensure that they do not affect the performance of other virtual machines and Containers on the Node.

By default, any newly created virtual machine or Container does not have the IOPS limit set and can perform so many disk I/O operations per second as necessary. To set the IOPS limit, you can use the `--iopslimit` option of the `prlctl set` command. For example, to allow Container 101 and the MyVM virtual machine to perform no more than 100 disk I/O operations per second, you can run the following commands:

```
# prlctl set 101 --iopslimit 100
# prlctl set MyVM --iopslimit 100
```

To ensure that the IOPS limit has been successfully applied to Container 101 and the `MyVM` virtual machine, use the `pstat -A` command:

```
# pstat -A
ST IOUSED IOWAIT IO          IOPS      NAME
OK   0.00   0.00  0.0/---KB/s 0.0/100/s MyVM
OK   0.00   0.00  0.0/---KB/s 0.0/100/s 101
```

The **IOPS** columns shows the IOPS limits currently applied to Container 101 and the `MyVM` virtual machine.

At any time, you can remove the set IOPS limits by running these commands:

```
# prlctl set 101 --iopslimit 0
# prlctl set MyVM --iopslimit 0
```

Viewing Disk I/O Statistics

In Parallels Containers 4.6, you can view disk input and output (I/O) statistics for virtual machines and Containers. To display the I/O statistics for all running virtual machines and Containers on the physical server, you can run the `pstat` utility with the `-a` option. For example:

```
# pstat -a
 7:18pm, up 1 day,  1:29,  2 users, load average: 0.00, 0.01, 0.00
CTNum 1, procs  127: R   2, S  125, D   0, Z   0, T   0, X   0
CPU [ OK ]: CTs   0%, CT0   1%, user   0%, sys   2%, idle  98%, lat(ms)  12/0
Mem [ OK ]: total 1560MB, free 627MB/402MB (low/high), lat(ms) 0/0
  ZONE0 (DMA): size 16MB, act 0MB, inact 0MB, free 11MB (0/0/0)
```

```
  ZONE1 (Normal): size 880MB, act 76MB, inact 104MB, free 616MB (3/4/5)
  ZONE2 (HighMem): size 684MB, act 116MB, inact 153MB, free 402MB (0/1/1)
  Mem lat (ms): A0 0, K0 0, U0 0, K1 0, U1 0
  Slab pages: 65MB/65MB (ino 43MB, de 9MB, bh 2MB, pb 0MB)
Swap [ OK ]: tot 2502MB, free 2502MB, in 0.000MB/s, out 0.000MB/s
Net [ OK ]: tot: in  0.005MB/s   45pkt/s, out  0.000MB/s    1pkt/s
             lo: in  0.000MB/s    0pkt/s, out  0.000MB/s    0pkt/s
           eth0: in  0.005MB/s   45pkt/s, out  0.000MB/s    1pkt/s
            br0: in  0.000MB/s    0pkt/s, out  0.000MB/s    0pkt/s
            br1: in  0.000MB/s    0pkt/s, out  0.000MB/s    0pkt/s
Disks [ OK ]: in 0.000MB/s, out 0.000MB/s

ST  IOUSED  IOWAIT           IO      IOPS  NAME
OK    0.00    0.00  0.0/---KB/s  0.0/--/s  101
OK    0.00    0.00  0.0/---KB/s  0.0/--/s  MyVM
```

The information related to the virtual machines and Containers disk I/O statistics is at the end of the command output. The table below explains the displayed I/O parameters:

Parameter	Description
IOUSED%	Percentage of time the disks are used by the virtual machine or Container.
IOWAIT%	Percentage of time when at least one I/O transaction in the virtual machine or Container is waiting for being served.
IO	I/O rate and limit, in bytes, kilobytes, megabytes, or gigabytes per second.
IOPS	I/O operations rate and limit, in operations per second.

The `pstat -a` command outputs the disk I/O statistics for all virtual machines and Containers that are currently running on the physical server. In the example output above, this statistics is shown for Container 101 and for the `MyVM` virtual machine.

Note: For more information on `pstat` and its options, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Setting Disk I/O Limits for Container Backups and Migrations

The operations of backing up, restoring, and migrating Containers can generate a high disk I/O load on the server, thus slowing down the performance of other Containers or of the server itself. You can avoid such situations by setting disk I/O limits for these operations.

To set a disk I/O limit, do the following:

1 Open the Parallels Cloud Server global configuration file for editing, for example:

```
# vi /etc/vz/vz.conf
```

2 Locate the following section in the file:

```
# VZ Tools IO limit
# To enable - uncomment next line, check the value - there should not be CT with the
same ID
# VZ_TOOLS_BCID=2
# Uncomment next line to specify required disk IO bandwidth in Bps (10485760 - 10MBps)
# VZ_TOOLS_IOLIMIT=10485760
```

3 Edit this section as follows:

a Uncomment the `VZ_TOOLS_BCID` parameter to enable disk I/O limits for backup, restore, and migration operations. When defining the parameter, make sure that no Container with the specified ID exists on the server.

b Uncomment the `VZ_TOOLS_IOLIMIT` parameter, and set the disk I/O limit for backup, restore, and migration operations. The value is set in bytes per second.

4 Save the file.

When setting disk I/O limits, pay attention to the following:

- `VZ_TOOLS_BCID` and `VZ_TOOLS_IOLIMIT` are global parameters—that is, once these parameters are set, they have effect on all Containers on the server.
- The `VZ_TOOLS_BCID` and `VZ_TOOLS_IOLIMIT` parameters control the disk I/O load only for backup, restore, and migration operations.

Locating Disk I/O Bottlenecks for Containers

In some situations, the disk I/O subsystem may bottleneck a Parallels Cloud Server server and reduce the performance of Containers or the server itself. Such I/O bottlenecks are often caused by disk-intensive processes running in Containers, for example, by transferring huge amounts of data to or from a Container.

To pinpoint Containers generating the highest disk I/O load, follow these steps:

1 On the problem server, run the `pstat` utility to display I/O statistics for all running Containers. You may want to use the following I/O options with the utility: `id`, `cpu_iowait`, `cpu_iowait_acc`, `io`, `iops`, `swapin`. For example:

```
# pstat -o id,cpu_iowait,cpu_iowait_acc,io,iops,swapin
...
CTID  CPU IOWAIT  ACC IOWAIT   IO           IOPS      SWAPIN
   1  0.00%       0.00%        0.0/---KB/s  0.0/--/s  0.0/s
 101  0.00%       0.00%        0.0/---KB/s  0.0/--/s  0.0/s
 111  25.0%       28.0%        23/---MB/s   419/--/s  0.0/s
```

2 In the command output, find the Container with the highest values. Most probably, this is the Container that generates the highest I/O load on the server. In the example above, this is Container 111.

3 Once you have found the problem Container, run the `vziotop -o -E <CT_ID>` command to show all threads running in this Container and generating I/O load. To sort the output by I/O load, use the left and right arrow keys to select the `IO` column. For example:

```
# vziotop -o -E 111
Total DISK READ :       4.77 M/s | Total DISK WRITE :      270.00 K/s
Actual DISK READ:       4.77 M/s | Actual DISK WRITE:      354.63 K/s
CTID  TID     PRIO  USER  DISK READ   DISK WRITE  SWAPIN  IO>    COMMAND
111   1026959 be/4  root  932.72 K/s  250.00 K/s  0.00%   42.84% {output skipped];
111   1026960 be/4  root  202.20 K/s  20.00  K/s  0.00%   11.53% {output skipped];
```

The command output shows that thread 1026959 produces the highest I/O load and might be responsible for the server performance degradation.

For more information on the `vziotop` utility and its options, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Managing Memory Parameters for Containers

This section describes the VSwap memory management system introduced in Parallels Cloud Server 6.0. You will learn to do the following:

- Configure the main VSwap parameters for Containers (p. 101).
- Set the memory allocation limit in Containers (p. 102).
- Configure OOM killer behavior (p. 103).
- Enhance the VSwap functionality (p. 104).

If you have upgraded to Parallels Cloud Server 6.0, you will also learn how the system calculates the new VSwap values (p. 104) from the memory parameters that were applied to Containers before the upgrade.

Configuring Main VSwap Parameters

Parallels Cloud Server 6.0 introduces a new scheme for managing memory-related parameters in Containers—`VSwap`. Like many other memory management schemes used on standalone Linux computers, this scheme is based on two main parameters:

- *RAM*. This parameter determines the total size of RAM that can be used by the processes of a Container.
- *swap*. This parameter determines the total size of swap that can be used by a Container for swapping out memory once the RAM is exceeded.

Notes:

1. In Parallels Cloud Server 6.0, the new VSwap memory management scheme has replaced the SLM scheme.

2. You can also set memory limits for and provide memory guarantees to Containers by configuring multiple UBC (User Beancounter) parameters (`numproc`, `numtcpsock`, `vmguarpages`, and so on). These parameters provide you with comprehensive facilities of customizing the memory resources in respect of your Containers. However, this way of managing system resources is more complex and requires more effort to be made on your part to adopt it to your system. For detailed information on UBC parameters, refer to the *Administrator's Guide to Managing UBC Resources* available at http://www.parallels.com/products/pvcl/resources/docs.

The new memory management scheme works as follows:

1 You set for a Container a certain amount of RAM and swap space that can be used by the processes running in the Container.

2. When the Container exceeds the RAM limit set for it, the swapping process starts.

   The swapping process for Containers slightly differs from that on a standalone computer. The Container swap file is virtual and, if possible, resides in the Node RAM. In other words, when the swap-out for a Container starts and the Node has enough RAM to keep the swap file, the swap file is stored in the Node RAM rather than on the hard drive.

3. Once the Container exceeds its swap limit, the system invokes the OOM Killer for this Container.

4. The OOM Killer chooses one or more processes running in the affected Container and forcibly kills them.

By default, any newly created Container starts using the new memory management scheme. To find out the amount of RAM and swap space set for a Container, you can check the values of the `PHYSPAGES` and `SWAPPAGES` parameters in the Container configuration file, for example:

```
# grep PHYSPAGES /etc/vz/conf/101.conf
PHYSPAGES="65536:65536"
# grep SWAPPAGES /etc/vz/conf/101.conf
SWAPPAGES="65536"
```

In this example, the value of the `PHYSPAGES` parameter for Container 101 is set to 65536. The `PHYSPAGES` parameter displays the amount of RAM in 4-KB pages, so the total amount of RAM set for Container 101 equals to 256 MB. The value of the `SWAPPAGES` parameter is also set to 256 MB.

To configure the amounts of RAM and swap space for Container 101, use the `--memsize` and `--swappages` options of the `prlctl set` command. For example, you can execute the following command to set the amount of RAM and SWAP in Container 101 to 1 GB and 512 MB, respectively:

```
# prlctl set 101 --memsize 1G --swappages 512M
```

Configuring the Memory Allocation Limit

When an application starts in a Container, it allocates a certain amount of memory for its needs. Usually, the allocated memory is much more than the application actually requires for its execution. This may lead to a situation when you cannot run an application in the Container even if it has enough free memory. To deal with such situations, the VSwap memory management scheme introduces the new vm`_overcommit` option. Using it, you can configure the amount of memory applications in a Container may allocate, irrespective of the amount of RAM and swap space assigned to the Container.

The amount of memory that can be allocated by applications of a Container is the sum of RAM and swap space set for this Container multiplied by a memory overcommit factor. In the default (basic) Container configuration file, this factor is set to 1.5. For example, if a Container is based on the default configuration file and assigned 1 GB of RAM and 512 MB of swap, the memory allocation limit for the Container will be 2304 MB. You can configure this limit and set it, for example, to 3 GB by running this command:

```
# prlctl set 101 --vm_overcommit 2
```

This command uses the factor of 2 to increase the memory allocation limit to 3 GB:

```
(1 GB of RAM + 512 MB of swap) * 2 = 3 GB
```

Now applications in Container 101 can allocate up to 3 GB of memory, if necessary.

Notes:

1. For more information on Container configuration files, see **Managing Container Resources Configurations** (p. 110).

2. Currently, the `--vm_overcommit` option may not be supported by the `prlctl` utility. Use `vzctl` instead.

Configuring OOM Killer Behavior

The OOM killer selects Container process (or processes) to end based on the badness reflected in `/proc/<pid>/oom_score`. The badness is calculated using process memory, total memory, and badness adjustment, and then clipped to the range from 0 to 1000. Each badness point stands for one thousandth of Container memory. The process to be killed is the one with the highest resulting badness.

The OOM killer for Container processes can be configured using the `/etc/vz/oom-groups.conf` file that lists patterns based on which badness adjustment is selected for each running process. Each pattern takes a single line and includes the following columns:

- *<command>* — mask for the task command name;
- *<parent>* — mask for the parent task name;
- *<oom_uid>* — task user identifier (UID) filter:
  - If *<oom_uid>* is -1, the pattern will be applicable to tasks with any UIDs,
  - If *<oom_uid>* is 0 or higher, the pattern will be applicable to tasks with UIDs equal to the *<oom_uid>* value,
  - If *<oom_uid>* is less than -1, the pattern will be applicable to tasks with UIDs less than *–<oom_uid>* (the negative *<oom_uid>* value);
- *<oom_score_adj>* — badness adjustment. As with badness itself, each adjustment point stands for one thousandth of total Container memory. Negative adjustment values reduce process badness. In an out-of-memory situation, an adjustment will guarantee that the process will be allowed to occupy at least *<oom_score_adj>* thousandths of Container memory while there are other processes with higher badness running in the Container.

Note: The *<command>* and *<parent>* masks support wildcard suffixes: asterisk (*) matches any suffix. E.g., "foo" matches only "foo", "foo*" matches "foo" and "foobar".

For example, the pattern

```
sshd     init     -500     -100
```

means that in an out-of-memory situation, `sshd`, a child of `init`, will be guaranteed at least 100 thousandths (i.e., 10%) of Container memory, if its UID is less than -(-500) or just 500, e.g., 499. According to RHEL conventions, UIDs from 1 to 499 are usually reserved for system use, so such delimitation may be useful to prioritize and save system processes.

While calculating the badness of a process, the OOM killer searches `/proc/vz/oom_score_adj` for a suitable pattern based on masks and task UID filter. The search starts from the first line and ends when the first suitable pattern is found. The corresponding adjustment value is then used to obtain the resulting process badness.

The data from `/etc/vz/oom-groups.conf` is reset and committed to the kernel on boot. To reset and commit the config file manually, you can use the following command:

```
# cat /etc/vz/oom-groups.conf > /proc/vz/oom_score_adj
```

Tuning VSwap

The new management scheme can be extended by using UBC parameters. For example, you can set the `numproc` parameter to configure the maximal number of processes and threads a Container may create or the `numfile` parameter to specify the number of files that may be opened by all processes in the Container. For detailed information on using UBC parameters, consult the *Administrator's Guide to Managing UBC Resources*.

Configuring Legacy Containers

If you upgrade from an earlier version of Parallels Cloud Server 6.0, all Containers start using the new memory management scheme after the upgrade. Every time a Container is started, the system automatically calculates the values of RAM, swap, and allocation limit from the memory parameters that were applied to a Container before the upgrade and uses them for the Container while it is running. The calculation rules are described below.

SLM

If a Container uses only SLM parameters:

- The amount of RAM is set to the value of the `SLMMEMORYLIMIT` parameter.
- The amount of swap is set to 0.
- The memory allocation limit is set to the value of the `SLMMEMORYLIMIT` parameter multiplied by the value of the `VM_OVERCOMMIT` parameter. By default, the `VM_OVERCOMMIT` parameter is not limited, which means that the memory allocation limit is also unlimited. To configure the limit for a Container, you need to set the `VM_OVERCOMMIT` parameter in its configuration file.

For example, if the `SLMMEMORYLIMIT` and `VM_OVERCOMMIT` parameters for Container 101 are set to 1 GB and 1.5, respectively, the Container will have them set to the following values after the upgrade: RAM = 1 GB, swap = 0, memory allocation limit = 1.5 GB.

UBC

If a Container uses only UBC parameters:

- The amount of RAM is set to the soft limit of the PRIVVMPAGES parameter.
- The amount of swap is set to 0.
- The memory allocation limit is set to the hard limit of the PRIVVMPAGES parameter

For example, if the soft limit of PRIVVMPAGES for Container 101 is set to 65536 pages and the hard limit to 131072, then the Container will have the following parameters: RAM = 256 MB, swap = 0, memory allocation limit = 2.

SLM and UBC

If a Container uses both SLM and UBC parameters:

- The amount of RAM is set to the value of the SLMMEMORYLIMIT parameter.
- The amount of swap is set to 0.
- The memory allocation limit is set to the value of the SLMMEMORYLIMIT parameter multiplied by multiplied by the value of the VM_OVERCOMMIT parameter. By default, the VM_OVERCOMMIT parameter is not limited, which means that the memory allocation limit is also unlimited. To configure the limit for a Container, you need to set the VM_OVERCOMMIT parameter in its configuration file.

For example, if the SLMMEMORYLIMIT and VM_OVERCOMMIT parameters for Container 101 are set to 1 GB and 1.5, respectively, the Container will have them set to the following values after the upgrade: RAM = 1 GB, swap = 0, memory allocation limit = 1.5 GB.

Managing Memory Parameters for Virtual Machines

This section describes the process of managing memory parameters for virtual machines. You will learn how to do the following:

- Configure the main and video memory for virtual machines (p. 105).
- Configure memory limits and quota for virtual machines (p. 106).
- Enable the memory hotplug functionality for virtual machines (p. 109).

Configuring Main Memory Parameters

The main memory resources for virtual machines include the following:

- main memory
- video memory

The process of managing these resources is described below.

Configuring Main Memory

To configure the amount of memory that will be available to the virtual machine, use the `--memsize` option of the `prlctl set` command. The following session shows how to change the amount of memory for the `MyVM` virtual machine from 512 MB to 756 MB and to check that the new value has been successfully set:

```
# prlctl list -i MyVM | grep memory
  memory 512Mb
# prlctl set MyVM --memsize 756
Set the memsize parameter to 756Mb
The VM has been successfully configured.
# prlctl list -i MyVM | grep memory
  memory 756Mb
```

You can configure the memory size for both running and stopped virtual machines.

Configuring Video Memory

To set the amount of video memory to be available to the virtual machine's video card, use the `--videosize` option of the `prlctl set` command. Assuming that the current video memory size of the `MyVM` virtual machine is set to 32 MB, you can increase it to 64 MB by running the following command:

```
# prlctl set MyVM --videosize 64
```

To check that the new value has been successfully set, use this command:

```
# prlctl list -i MyVM | grep video
  video 64Mb
```

Configuring Additional Memory Parameters

Parallels Cloud Server 6.0 comes with an improved memory management scheme. This scheme is driven by two parameters:

- *Reserved memory limit*. The reserved memory limit defines the amount of memory on a Parallels server that can be used by all virtual machines hosted on this server.
- *Memory quota*. The memory quota controls the memory consumption by a particular virtual machine. This parameter is composite and includes the guarantee, limit, priority, and ballooning settings.

The sections below describe how to configure both parameters.

Configuring the Reserved Memory Limit

The reserved memory limit defines the amount of memory that can be consumed by all virtual machines on a Parallels server. The remaining memory on the server is reserved for applications that run on the server itself.

By default, the reserved memory limit is calculated automatically and depends on the amount of memory installed on a Parallels server:

- If the server has less than 3 GB of memory installed, this formula is used: "total RAM on the server" multiplied by 0.7. So if the server has 2 GB of memory, the reserved limit is set to 1.4 GB.
- If the server has more than 3 GB of memory installed, this formula is used: "total RAM on the server" minus 1 GB. So if the server has 16 GB of memory, the reserved limit is set to 15 GB.

To configure the default reserved memory limit, you can use the `--mem-limit` option of the `prlsrvctl set` command. For example, the following command reserves 14 GB of memory for use by virtual machines:

```
# prlsrvctl set --mem-limit 14336
```

To revert to the default settings, use this command:

```
# prlsrvctl set --mem-limit auto
Set memory limit: auto
The Server has been successfully configured.
```

Configuring the Memory Quota

The memory quota allows you to control the memory consumption by a particular virtual machine. The quota control parameters include the following:

- *Guarantee*. The amount of memory a virtual machine is guaranteed to get on demand. If the virtual machine requests more memory than is guaranteed, the memory allocation may fail (for example, if there is no free memory on the Parallels server at the moment). Moreover, if the guaranteed amount of memory of all virtual machines running on the server plus their overhead exceeds the reserved limit, you will not be able to start another virtual machine. By default, the guaranteed memory is calculated on the basis of RAM and video memory assigned to a virtual machine and is about a half of its total memory.

  Note: The overhead of a virtual machine depends on its configuration. For example, the overhead of a virtual machine that has 1024 MB of RAM, 2 CPUs, 256 MB of video memory, a network adapter, a sound card, and a USB controller and runs on modern hardware does not usually exceed 35-40 MB. To check the overhead of a running virtual machine, open the `/proc/parallels/vm/`*`VM_ID`*`/meminfo` file and look for the value of the `Unreclaimable` parameter. Keep in mind, however, that this value may change over time.

- *Limit*. The maximum amount of memory a virtual machine is allowed to consume. The virtual machine cannot exceed this limit even if the Parallels server has a lot of free memory and the virtual machine requires this memory. By default, no limit is set for all newly created virtual machines, and any virtual machine may consume all free memory on the server.
- *Priority*. The priority (from 1 to 100) that defines which virtual machine will get memory first. The higher the priority of a virtual machine, the more chances it has to get memory when the Parallels server has insufficient memory resources. By default, the priority is set to 50.
- *Ballooning*. The maximum amount of memory the balloon driver in a virtual machine may allocate for its needs. Memory ballooning is a technique that allows your system to reclaim memory from virtual machines. To do this, a special balloon driver is loaded into each running

virtual machine. When the system requires free memory but does not have any, it sends a command to the balloon driver in the virtual machine to increase its size. The balloon driver inflates by allocating the requested amount of memory in the virtual machine and then gives this memory to the system.

By default, the balloon driver can allocate up to 60% of RAM set for a virtual machine. For example, if the amount of RAM for a virtual machine is set to 2 GB, the balloon driver can allocate the maximum of 1.2 GB of memory.

To configure these quota control parameters for a specific virtual machine, you can use the `--memquota` parameter of the `prlctl set` command. For example, the following command sets for the `MyVM` virtual machine the memory guarantee to 512 MB, the limit to 2 GB, the priority to 70, and the ballooning limit to 50% of RAM:

```
# prlctl set MyVM --memquota 512:2048:70:50
```

To check that all parameters have been successfully set, use this command:

```
# prlctl list -i MyVM | grep memory_quota
memory_quota  min=512Mb max=2048Mb priority=70 maxballoon=50%
```

To revert to the default settings, run this command:

```
# prlctl set MyVM --memquota auto
```

Possible Usage Scenarios

Below you can find three examples that demonstrate how the new management scheme can help service providers optimize their business.

"Ghetto" (payable RAM)

You have a number of non-priority customers whose virtual machines use only a fraction of the RAM assigned to them. The virtual machines are stored on different physical servers. To optimize the resources usage, you decide to migrate all underutilized virtual machines to one "Ghetto" server. You then set (a) the guarantee for the migrated virtual machines to the minimum values recommended for the operating systems running in these VMs plus the VM overhead and (b) the memory limit to their RAM values. In this scenario:

- You host a large number of virtual machines on a single physical server.
- You ensure that all virtual machines can get enough memory for non-intensive operations, but do not guarantee the optimal performance of memory-intensive operations (the limit is equal to the RAM).
- You charge customers for the amount of RAM assigned to their virtual machines.

"What you pay is what you get" (payable guarantee and limit)

You have a number of customers whose virtual machines require a certain amount of memory all the time. For these customers, you configure their virtual machines by setting the memory guarantee to the requested amount plus the VM overhead. You also set the memory limit equal to the guarantee. In this scenario:

- You charge customers for the set memory guarantee.

- Customers can get only the memory they pay for (the guarantee is equal to the limit).
- You ensure that every virtual machine can get the required amount of memory defined by its guarantee.
- No virtual machine can affect the performance of the host and other virtual machines on this host. To meet this requirement, you need to make sure that all virtual machines on the host are configured for use in the "What you pay is what you get" scenario.

"Burstable memory" (payable guarantee)

You have a number of customers whose virtual machines consume small amounts of memory most of the time. Sometimes, memory-intensive operations may also run in the virtual machines. For these virtual machines, you set the memory guarantee to the values that are high enough to run non-intensive memory operations and the memory limit to "unlimited". In this scenario:

- You charge customers for the set memory guarantee.
- You ensure that all virtual machines can get enough memory for non-intensive operations, and such operations are running in the affected virtual machines most of the time.
- When virtual machines require more memory than is defined by their guarantee, they get free memory available on the host (the limit is set to "unlimited"). If the host does not have enough memory, the virtual machines start competing for free memory.

Enabling Memory Hotplug for Virtual Machines

If a guest operating system supports the memory hotplug functionality, you can enable this functionality for the virtual machine. Once the memory hotplug functionality is turned on, you can increase the amount of memory available to your virtual machines even if they are running.

Note: Decreasing the amount of memory available to a running virtual machine is not supported in the current version of Parallels Cloud Server.

Currently, the following systems come with the memory hotplug support:

Linux (both x86 and x64 versions)

- CentOS 5.3 and higher
- Red Hat Enterprise LInux 5.3 and higher
- Fedora 13 and higher
- SUSE Linux Enterprise Server 10 and higher
- Ubuntu 10.04 and higher

Windows

- x64 version of Windows Server 2008 R2 (Datacenter Edition)
- x86 and x64 versions of Windows Server 2008 (Standard Edition)
- x86 and x64 versions of Windows Server 2008 (Enterprise Edition)

- x86 and x64 versions of Windows Server 2008 (Datacenter Edition)
- x86 and x64 versions of Windows Server 2003 (Enterprise Edition)

By default, memory hotplug support is disabled for all newly created virtual machines. To enable this functionality, you can use the `--mem-hotplug` option of the `prlctl set` command. For example, to enable the memory hotplug support in the `MyVM` virtual machine that runs one of the supported operating systems, stop the `MyVM` virtual machine and run this command:

```
# prlctl set MyVM --mem-hotplug on
set mem hotplug: 1
The VM has been successfully configured.
```

Once the functionality is enabled, you can increase the amount of memory for the `MyVM` virtual machine even it is running.

To disable the memory hotplug support in the `MyVM` virtual machine, use this command:

```
# prlctl set MyVM --mem-hotplug off
set mem hotplug: 0
The VM has been successfully configured.
```

The changes will come into effect on the next virtual machine start.

Managing Container Resources Configuration

Any Container is configured by means of its own configuration file. You can manage Container configurations in a number of ways:

1. Using configuration sample files shipped with Parallels Cloud Server. These files are used when a new Container is being created (for details, see **Creating and Configuring New Containers**). Currently, the following configuration sample files are provided:
   - `basic`. Use it for creating standard Containers.
   - `confixx`. Use it for creating Containers that are to run the Confixx control panel.
   - `vswap.plesk`. Use it for creating Containers with the Plesk control panel.
   - `vswap.256MB`. Use it for creating Containers with 256 MB of main memory.
   - `vswap.512Mb`. Use it for creating Containers with 512 MB of main memory.
   - `vswap.1024Mb`. Use it for creating Containers with 1024 MB of main memory.
   - `vswap.2048Mb`. Use it for creating Containers with 2048 MB of main memory.

   Note: Configuration sample files cannot contain spaces in their names.

   Any sample configuration file can also be applied to an existing Container. You would do this if, for example, you want to upgrade or downgrade the overall resources configuration of a particular Container:

```
# prlctl set 101 --applyconfig basic
```

   This command applies all the parameters from the `ve-basic.conf-sample` file to Container

101.

When you install Parallels Cloud Server on your server, the default Container samples are put to the `/etc/vz/conf` directory. They have the following format: `ve-<name>.conf-sample` (for example, `ve-basic.conf-sample`). In this connection you should keep in mind the following when working with Container samples:

- When you create a Container using the `prlctl create` command utility and base it on some Container sample, this sample is taken from the `/etc/vz/conf` directory.

2 Using specific utilities for preparing configuration files in their entirety. The tasks these utilities perform are described in the following subsections of this section.

3 The direct creating and editing of the corresponding Container configuration file (`/etc/vz/conf/<CT_ID>.conf`). This can be performed with the help of any text editor. The instructions on how to edit Container configuration files directly are provided in the four preceding sections. In this case you have to edit all the configuration parameters separately, one by one.

Splitting Server Into Equal Pieces

It is possible to create a Container configuration roughly representing a given fraction of the server. If you want to create such a configuration that up to 20 fully loaded Containers would be able to be simultaneously running on the given server, you can do it as follows:

```
# cd /etc/vz/conf
# vzsplit -n 20 -f mytest
Config /etc/vz/conf/ve-mytest.conf-sample was created
```

Notice that the configuration produced depends on the given server resources. Therefore, it is important to validate the resulted configuration file before trying to use it, which is done with the help of the `vzcfgvalidate` utility. For example:

```
# vzcfgvalidate ve-mytest.conf-sample
Validation completed: success
```

The command output shows that the configuration file is valid and you can safely use as the basis for creating new Containers.

Note: If you generate a Container configuration sample using the `vzsplit` command line utility, the resulting Container sample is put to the `/etc/vz/conf` directory. This sample can then be used by `prlctl create` when creating a new Container on its basis.

Scaling Container Configuration

Any configuration or configuration sample file can prove insufficient for your needs. You might have an application which does not fit into existing configurations. The easiest way of producing a Container configuration is to scale an existing one.

Scaling produces a “heavier” or “lighter” configuration in comparison with an existing one. All the parameters of the existing configuration are multiplied by a given number. A heavier configuration is produced with a factor greater than 1, and a lighter one – with a factor between 0 and 1.

Note: If you create a new sample on the basis of an existing sample using the `vzcfgscale` command line utility, the resulting Container sample is put to the `/etc/vz/conf` directory. This sample can then be used by `prlctl create` when creating a new Container on its basis.

The session below shows how to produce a configuration sample 50% heavier than the `basic` configuration shipped with Parallels Cloud Server:

```
# cd /etc/vz/conf
# vzcfgscale -a 1.5 -o ve-improved.conf-sample ve-basic.conf-sample
# vzcfgvalidate ve-improved.conf-sample
Validation completed: success
```

Now you can use the `improved` configuration file to create new Containers.

It is possible to use the same technique for scaling configurations of the existing Containers. Notice that the output file cannot be the same as the file being scaled. You have to save the scaling results into an intermediate file.

Applying New Configuration Sample to Container

Parallels Cloud Server allows you to change the configuration sample file a Container is based on and, thus, to modify all the resources the Container may consume and/or allocate at once. For example, if Container 101 is currently based on the `basic` configuration sample and you are planning to run the Plesk application inside the Container, you may wish to apply the `vswap.plesk` sample to it instead of `basic`, which will automatically adjust the necessary Container resource parameters for running the Plesk application inside Container 101. To do this, you can execute the following command on the server:

```
# prlctl set 101 --applyconfig vswap.plesk
```

This command reads the resource parameters from the `ve-vswap.plesk.conf-sample` file located in the `/etc/vz/conf` directory and applies them one by one to Container 101.

When applying new configuration samples to Containers, keep in mind the following:

- All Container sample files are located in the `/etc/vz/conf` directory on the server and are named according to the following pattern: `ve-<name>.conf-sample`. You should specify only the *<name>* part of the corresponding sample name after the `--applyconfig` option (`vswap.plesk` in the example above).
- The `--applyconfig` option applies all the parameters from the specified sample file to the given Container, except for the `OSTEMPLATE`, `TEMPLATES`, `VE_ROOT`, `VE_PRIVATE`, `HOSTNAME`, `IP_ADDRESS`, `TEMPLATE`, `NETIF` parameters (if they exist in the sample file).

You may need to restart your Container depending on the fact whether the changes for the selected parameters can be set on the fly or not. If some parameters could not be configured on the fly, you will be presented with the corresponding message informing you of this fact.

Managing Virtual Machine Configuration Samples

The configuration of a virtual machine is defined by its `config.pvs` configuration file. This file in XML format is automatically created when you make a new virtual machine and contains all parameters of the virtual machine: memory, CPU, disk space, and so on.

Once a virtual machine is created, you can manually configure its parameters using the `prlctl` utility. However, if you need to configure multiple parameters for several virtual machines, this may become a tedious task. To facilitate your work, you can create *virtual machine samples* and use them to quickly and easily change the configuration of virtual machines. You can even further simplify the configuration process by creating a virtual machine template and several sample files. In this case, you can quickly make a new virtual machine on the basis of your template and apply the desired configuration file to it.

Creating a Configuration Sample

Before you can start using virtual machine configuration samples, you need to create at least one configuration sample. The easiest way of doing this is to follow the steps below:

1. Create a virtual machine configuration, for example:

```
# prlctl create VmConfiguration
```

2. Set the parameters for the virtual machine configuration as you want them to be. For example, you can use the `prlctl set` command to set the required amount of memory and disk space. All your parameters are saved to the `config.pvs` file of the `VmConfiguration` virtual machine.

   For the list of parameters that can be applied from a configuration sample, see **Parameters Applied from Configuration Samples** below.

3. Copy the `config.pvs` file to the `/etc/parallels/samples` directory. If this directory does not exist, create it:

```
# mkdir /etc/parallels/samples
# cp /var/parallels/VmConfiguration/config.pvs /etc/parallels/samples/configMySQL.pvs
```

The latter command copies the `config.pvs` file to the `configMyDB.pvs` file.

Applying Configuration Samples to Virtual Machines

Now that you have created the configuration sample, you can apply it to any of your virtual machines. You can do this using the `--applyconfig` option with the `prlctl set` command and specifying the sample name without the `.pvs` extension. For example, to apply the `configMySQL` sample to the `VM1` virtual machine, you can run this command:

```
# prlctl set VM1 --applyconfig configMySQL
```

You can apply configuration samples to stopped virtual machines only.

Parameters Applied from Configuration Samples

The following parameters are applied to a virtual machine from a new configuration sample:

- All memory-related parameters (both RAM and video). To view these parameters in a sample file, locate the `<Memory>` and `<Video>` elements.
- All CPU-related parameters. To view these parameters in a sample file, locate the `<Cpu>` element.
- IO and IOPS parameters. To view these parameters in a sample file, locate the `<IoLimit>` and `<IopsLimit>` elements, respectively.
- Disk space parameter. To view this parameter in a sample file, locate the `<Size>` element enclosed in the `<Hdd>` element:

```
<Hdd id=0" dyn_lists="Partition 0">
   <Index>0</Index>
   <Size>65536</Size>
</Hdd>
```

  The virtual disk to which the value of the `<Size>` element is applied is defined by the index number in the `<Index>` element. For example, in the example above, the disk space parameter (65536 MB) is applied to the virtual disk with index number 0. If the virtual machine does not have a virtual disk with the specified index, the parameter is ignored.

Monitoring Resources

In Parallels Cloud Server, you can use the `pstat` utility to monitor system resources in real time. When executed, the utility displays the status and load of the system: its disk, network, CPU, memory, and other parameters. It also provides the list of running virtual machines and Containers together with their resources consumption statistics. For example, you can run the following command on the server to view your current system resources:

```
# pstat -d 5
   5:39pm, up 4 days,  5:33,  2 users, load average: 1.08, 1.11, 1.05
CTNum 2, procs  268: R   1, S  265, D   1, Z   0, T   1, X   0
CPU [ OK ]: CTs   0%, CT0   0%, user   0%, sys   1%, idle  99%, lat(ms)   1/0
Mem [ OK ]: total 7831MB, free 4147MB/0MB (low/high), lat(ms) 1/0
  ZONE0 (DMA): size 9MB, act 0MB, inact 0MB, free 10MB (0/0/0)
  ZONE1 (DMA32): size 3238MB, act 42MB, inact 39MB, free 3118MB (4/5/6)
  ZONE2 (Normal): size 4661MB, act 2730MB, inact 606MB, free 1019MB (6/8/9)
  Mem lat (ms): A0 0, K0 1, U0 0, K1 1, U1 0
  Slab pages: 181MB/181MB (ino 39MB, de 13MB, bh 21MB, pb 40MB)
Swap [ OK ]: tot 2000MB, free 2000MB, in 0.000MB/s, out 0.000MB/s
Net [ OK ]: tot: in  0.027MB/s  233pkt/s, out  0.040MB/s   37pkt/s
             lo: in  0.000MB/s    0pkt/s, out  0.000MB/s    0pkt/s
           eth0: in  0.014MB/s  116pkt/s, out  0.020MB/s   19pkt/s
           sit0: in  0.000MB/s    0pkt/s, out  0.000MB/s    0pkt/s
            br0: in  0.000MB/s    0pkt/s, out  0.000MB/s    0pkt/s
            br1: in  0.013MB/s  116pkt/s, out  0.020MB/s   19pkt/s
```

```
Disks [ OK ]: in 0.000MB/s, out 0.000MB/s

ST    %VM     %KM      CPU  FCNT MLAT NAME
OK 0.0/27  0.0/-   0.00/33     0    0 1
OK 0.2/685 0.0/-   0.00/33     0    0 101
OK 0.4/685 0.0/-   0.00/33     0    0 102
OK  27/6.7 0.1/-   0.00/33     0    0 Windows7
```

The command output is updated with the time interval equal to the value specified after the `-d` (delay) option measured in seconds. In the session above, the statistics displayed is renewed every five seconds. If the `-d` option is not specified, the default interval equals 1 second.

As you can see, the utility provides real-time information on all main resources subsystems pertaining both to the physical server and to its virtual machines and Containers: the disk, network, CPU, and memory subsystems. You may want to shrink the output of the utility by specifying the `-b` (brief) option instead of the `-v` (verbose) one, or to do without any options to use the "normal" mode of displaying.

The following information is displayed by default per each virtual machine or Container:

Column	Description
ST	virtual machine or Container status. If there are no failed counters and the latency values are normal, the status is "OK". Otherwise, it is displayed in red as "!!". You can sort virtual machines and Containers by their status to see the problem virtual machines and Containers first.
%VM	Virtual memory usage, in per cent to the total memory. The first number is how much virtual memory is being used, and the second one is the virtual memory barrier.
%KM	Kernel memory usage, in per cent to the normal zone size. The first number is how much kernel memory is being used, and the second one is the kernel memory barrier.
CPU	CPU usage in per cent to all available CPUs. The first number is how much of the CPU power is being used by the virtual machine or Container, and the second one is its guaranteed share judging by the `cpuunits` parameter. Note that the actual CPU usage may be higher than the guaranteed one.
FCNT	The number of failed counters for all the resource parameters. In the standard mode of displaying, this number represents the increase of failed counters since the previous screen update, whereas in the average mode of displaying, it represents an absolute failed counters sum for the given virtual machine or Container.
MLAT	Maximal scheduling latency for the virtual machine or Container, in ms. This parameter shows the maximal scheduling latency inside the given virtual machine or Container, i.e. for how long (at the utmost) a process inside the virtual machine or Container awaits for the CPU.
NAME	Virtual machine or Container name.

The **%VM**, **%KM**, and **CPU** columns provide two values per column separated by a slash for each virtual machine and Container. The first value indicates the real usage of the corresponding parameter by the virtual machine and Container, and the second one – the maximal value allowed for the virtual machine and Container.

For detailed information on options that you can use with the `pstat` utility, consult the *Parallels Cloud Server Command Line Guide*.

CHAPTER 5

Managing Services and Processes

This chapter provides information on what services and processes are, how they influence the operation and performance of your system, and what tasks they perform in the system.

You will learn how to use the command line utilities in order to manage services and processes in Parallels Cloud Server. In particular, you will learn how to monitor active processes in your system, change the mode of the `xinetd`-dependent services, identify the Container ID where a process is running by the process ID, start, stop, or restart services and processes, and edit the service run levels.

Note: In the current version of Parallels Cloud Server, you cannot use Parallels Cloud Server utilities to manage services and processes in virtual machines. However, you can log in to a particular virtual machine (e.g., via RDP to a Windows virtual machine and SSH to a Linux virtual machine) and manage its services and processes in the same way you would manage them on a standalone computer.

In This Chapter

What Are Services and Processes 116
Main Operations on Services and Processes..... 117
Managing Processes and Services 118

What Are Services and Processes

Instances of any programs currently running in the system are referred to as processes. A process can be regarded as the virtual address space and the control information necessary for the execution of a program. A typical example of a process is the *vi* application running on your server or inside your Linux-based Containers. Along with common processes, there are a great number of processes that provide an interface for other processes to call. They are called services. In many cases, services act as the brains behind many crucial system processes. They typically spend most of their time waiting for an event to occur or for a period when they are scheduled to perform some task. Many services provide the possibility for other servers on the network to connect to the given one via various network protocols. For example, the `nfs` service provides the NFS server functionality allowing file sharing in TCP/IP networks.

You may also come across the term "daemon" that is widely used in connection with processes and services. This term refers to a software program used for performing a specific function on the server system and is usually used as a synonym for "service". It can be easily identified by "d" at the end of its name. For example, `httpd` (short for the HTTP daemon) represents a software program that runs in the background of your system and waits for incoming requests to a web server. The

daemon answers the requests automatically and serves the hypertext and multimedia documents over the Internet using HTTP.

When working with services, you should keep in mind the following. During the lifetime of a service, it uses many system resources. It uses the CPUs in the system to run its instructions and the system's physical memory to hold itself and its data. It opens and uses files within the file systems and may directly or indirectly use certain physical devices in the system. Therefore, in order not to decrease your system performance, you should run only those services on the Parallels server that are really needed at the moment.

Besides, you should always remember that running services in the Host OS is much more dangerous than running them in virtual machines and Containers. In case violators get access to one of the virtual machines and Containers through any running service, they will be able to damage only the virtual machine and Container where this service is running, but not the other virtual machines and Containers on your server. The Parallels server itself will also remain unhurt. And if the service were running on the Parallels server, it would damage both the server and all virtual machines and Containers residing on it. Thus, you should make sure that you run only those services on the server that are really necessary for its proper functioning. Launch all additional services you need at the moment inside separate virtual machines and Containers. It can significantly improve your system safety.

Main Operations on Services and Processes

The ability to monitor and control processes and services in your system is essential because of the profound influence they have on the operation and performance of your whole system. The more you know about what each process or service is up to, the easier it will be to pinpoint and solve problems when they creep in.

The most common tasks associated with managing services running on the Parallels server or inside a virtual machine and Container are starting, stopping, enabling, and disabling a service. For example, you might need to start a service in order to use certain server-based applications, or you might need to stop or pause a service in order to perform testing or to troubleshoot a problem.

For `xinetd`-dependent services, you do not start and stop but enable and disable services. The services enabled in this way are started and stopped on the basis of the corresponding state of the `xinetd` daemon. Disabled services are not started whatever the `xinetd` state.

In Parallels Cloud Server, you can manage services on the Parallels server and inside Containers by means of special Linux command-line utilities. You can do it either locally or from any server connected on the network.

As for processes, such Parallels Cloud Server utilities as `vzps`, `vztop`, `vzpid` enable you to see what a process is doing and to control it. Sometimes, your system may experience problems such as slowness or instability, and using these utilities can help you improve your ability to track down the causes. It goes without saying that in Parallels Cloud Server you can perform all those

operations on processes you can do in a normal system, for example, kill a process by sending a terminate signal to it.

Managing Processes and Services

In Parallels Cloud Server, services and processes can be managed using the following Parallels command line utilities:

- `vzps`
- `vzpid`
- `vztop`

With their help, you can perform the following tasks:

- print the information about active processes on your Parallels server
- view the processes activity in real time
- change the mode of the services that can be either `xinetd`-dependent or standalone
- identify the Container ID where a process is running by the process ID

Note: In the current version of Parallels Cloud Server, you cannot use Parallels Cloud Server utilities for managing services and processes in virtual machines. However, you can log in to a particular virtual machine (e.g. via RDP to a Windows virtual machine and SSH to a Linux virtual machine) and manage its services and processes in the same way you would manage them on a standalone computer.

Viewing Active Processes and Services

The `vzps` utility provides certain additional functionality related to monitoring separate Containers running on the Parallels server. For example, you can use the `-E` switch with the `vzps` utility to:

- display the Container IDs where the processes are running
- view the processes running inside a particular Container

`vzps` prints the information about active processes on your Parallels server. When run without any options, `vzps` lists only those processes that are running on the current terminal. Below is an example output of the `vzps` run:

```
# vzps
  PID TTY          TIME CMD
 4684 pts/1    00:00:00 bash
27107 pts/1    00:00:00 vzps
```

Currently, the only processes assigned to the user/terminal are the `bash` shell and the `vzps` command itself. In the output, the PID (Process ID), TTY, TIME, and CMD fields are contained. TTY denotes which terminal the process is running on, TIME shows how much CPU time the process has used, and CMD is the name of the command that started the process.

Note: The IDs of the processes running inside Containers and displayed by running the `vzps` command on the Parallels server does not coincide with the IDs of the same processes shown by running the `ps` command inside these Containers.

As you can see, the standard `vzps` command just lists the basics. To get more details about the processes running on your server, you will need to pass some command line arguments to `vzps`. For example, using the `aux` arguments with this command displays processes started by other users (`a`), processes with no terminal or one different from yours (`x`), the user who started the process and when it began (`u`).

```
# vzps aux
USER   PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root     1  0.0  0.0   1516   128 ?        S    Jul14   0:37 init
root     5  0.0  0.0      0     0 ?        S    Jul14   0:03 [ubstatd]
root     6  0.0  0.0      0     0 ?        S    Jul14   3:20 [kswapd]
#27      7  0.0  0.0      0     0 ?        S    Jul14   0:00 [bdflush]
root     9  0.0  0.0      0     0 ?        S    Jul14   0:00 [kinoded]
root  1574  0.0  0.1    218   140 pts/4    S    09:30   0:00 -bash
```

There is a lot more information now. The fields USER, %CPU, %MEM, VSZ, RSS, STAT, and START have been added. Let us take a quick look at what they tell us.

The USER field shows you which user initiated the command. Many processes begin at system start time and often list root or some system account as the USER. Other processes are, of course, run by individuals.

The %CPU, %MEM, VSZ, and RSS fields all deal with system resources. First, you can see what percentage of the CPU the process is currently utilizing. Along with CPU utilization, you can see the current memory utilization and its VSZ (virtual memory size) and RSS (resident set size). VSZ is the amount of memory the program would take up if it were all in memory; RSS is the actual amount currently in memory. Knowing how much a process is currently eating will help determine if it is acting normally or has spun out of control.

You will notice a question mark in most of the TTY fields in the `vzps aux` output. This is because most of these programs were started at boot time and/or by initialization scripts. The controlling terminal does not exist for these processes; thus, the question mark. On the other hand, the `bash` command has a TTY value of pts/4. This is a command being run from a remote connection and has a terminal associated with it. This information is helpful for you when you have more than one connection open to the machine and want to determine which window a command is running in.

STAT shows the current status of a process. In our example, many are sleeping, indicated by an S in the STAT field. This simply means that they are waiting for something. It could be user input or the availability of system resources. The other most common status is R, meaning that it is currently running.

Note: For detailed information on all `vzps` parameters, output fields, states of processes, etc., please consult the `vzps` manual pages.

You can also use the `vzps` command to view the processes inside any running Container. The example below shows you how to display all active processes inside Container 101:

```
# vzps -E 101
CTID   PID TTY          TIME CMD
101 27173 ?        00:00:01 init
101 27545 ?        00:00:00 syslogd
101 27555 ?        00:00:00 sshd
101 27565 ?        00:00:00 xinetd
101 27576 ?        00:00:03 httpd
101 27583 ?        00:00:00 httpd
101 27584 ?        00:00:00 httpd
101 27587 ?        00:00:00 crond
101 27596 ?        00:00:00 saslauthd
```

Monitoring Processes in Real Time

The `vztop` utility is rather similar to `vzps` but is usually started full-screen and updates continuously with process information. This can help with programs that may infrequently cause problems and can be hard to see with `vzps`. Overall system information is also presented, which makes a nice place to start looking for problems.

The `vztop` utility can be run on the server just as the standard Linux `top` utility. The only features that distinguish the `vztop` utility from `top` are the following:

- `vztop` allows you to use the `-E` option that monitors only the processes belonging to the Container whose processes you want to display.
- You can use the `e` interactive command to temporarily view/hide the CTIDs where the processes are running.
- You can use the `E` interactive command to set the filter on the CTID field that helps you display only the processes belonging to the given Container.

The `vztop` utility usually has an output like the following:

```
# vztop -E 101
17:54:03  up 20 days, 23:37, 4 users, load average: 2.13, 1.89, 1.75
305 processes: 299 sleeping, 3 running, 3 zombie, 0 stopped
CPU0 states: 20.1% user 51.2% system  0.0% nice 0.0% iowait 28.1% idle
CPU1 states: 21.2% user 50.0% system  0.0% nice 0.0% iowait 28.1% idle
Mem: 1031088k av, 969340k used, 61748k free, 0k shrd, 256516k buff
       509264k active,              330948k inactive
Swap: 4056360k av,   17156k used, 4039204k free      192292k cached
CTID   PID USER PR  NI  VIRT  RES  SHR S %CPU %MEM  TIME+  COMMAND
101  27173 root 16   0  1616  604 1420 S  0.0  0.1  0:01.86 init
101  27545 root 16   0  1520  624 1356 S  0.0  0.1  0:00.34 syslogd
101  27555 root 25   0  4008 1700 3632 S  0.0  0.4  0:00.04 sshd
101  27565 root 25   0  2068  860 1740 S  0.0  0.2  0:00.05 xinetd
101  27576 root 16   0  7560 3180 6332 S  0.0  0.7  0:03.78 httpd
101  27587 root 16   0  2452 1036 1528 S  0.0  0.2  0:00.34 crond
101  27596 root 25   0  4048 1184 3704 S  0.0  0.2  0:00.01 saslauthd
```

As you can see, `vztop` provides an ongoing look at the processor activity in real time (the display is updated every 5 seconds by default, but you can change that with the `d` command-line option or the `s` interactive command). It displays a list of the most CPU-intensive tasks on the system and can provide an interactive interface for manipulating processes. It can sort the tasks by CPU usage, memory usage, and runtime. Specifying 101 after the `-E` option allows you to display only those

processes that are running inside Container 101 only. Besides, most features can be selected by an interactive command, for example, the `e` and `E` commands described above.

Note: In the current version of Parallels Cloud Server, you cannot use the `vztop` utility for monitoring processes in virtual machines.

Determining Container Identifiers by Process IDs

Each process is identified by a unique PID (process identifier), which is the entry of that process in the kernel's process table. For example, when you start Apache, it is assigned a process ID. This PID is then used to monitor and control this program. The PID is always a positive integer. In Parallels Cloud Server, you can use the `vzpid` (retrieve process ID) utility to print the Container ID the process with the given id belongs to. Multiple process IDs can be specified as arguments. In this case the utility will print the Container number for each of the processes.

The typical output of the `vzpid` utility is shown below:

```
# vzpid 12
Pid    VEID    Name
12      101    init
```

In our example the process with the identifier 12 has the name 'init' and is running in the Container with ID 101.

Note: You can also display the Container ID where the corresponding process is running by using the `vzps` utility.

CHAPTER 6

Managing Parallels Cloud Server Network

The given chapter familiarizes you with the Parallels Cloud Server network structure, enumerates Parallels networking components, and explains how to manage these components in your working environments. In particular, it provides the following information:

- How you can manage network adapters on the Parallels server.
- What Virtual Networks are and how you can manage them on the Parallels server.
- How to create virtual network adapters inside your virtual machines and Containers and configure their parameters.
- How to connect virtual machines and Containers to different networks.

In This Chapter

Managing Network Adapters on the Parallels Server 122
Networking Modes in Parallels Cloud Server 124
Configuring Virtual Machines and Containers in Host-Routed Mode 131
Configuring Virtual Machines and Containers in Bridged Mode 133
Managing Private Networks 143
Configuring Offline Management 148
Using Open vSwitch Bridges 154

Managing Network Adapters on the Parallels Server

Network adapters installed on the Parallels server are used to provide virtual machines and Containers with access to each other and to external networks. During the installation, Parallels Cloud Server registers all physical and VLAN network adapters available on the server. In addition to that, it creates a number of VLAN adapters on the server. Once Parallels Cloud Server has been successfully installed, you can perform the following operations on network adapters:

- List the adapters currently available on the server.
- Create new VLAN adapters on the server.

> **Note:** For more information on Virtual Networks, refer to **Managing Virtual Networks** (p. 133).

These operations are described in the following subsections in detail.

Listing Adapters

You can view the physical, virtual, and VLAN network adapters on your Parallels server using the `vznetcfg` utility. For example, you can execute the following command to list the available adapters:

```
# vznetcfg if list
Name    Type     Network ID  Addresses
eth0    nic      Bridged     10.30.18.41/16,fe80::20c:29ff:fee8:9419/64,dhcp
br1     bridge   Bridged     fe80::200:ff:fe00:0/64
br0     bridge   Host-Only   fe80::200:ff:fe00:0/64
```

The information on adapters is presented in the table with the following columns:

Column	Description
`Name`	Adapter name.
`Type`	Type of the network adapter. It can be one of the following: • `nic` denotes a physical adapter. • `vlan` stands for a VLAN adapter. • `bridge` is a virtual bridge automatically created for each Virtual Network on the Parallels server. • `veth`*X* is a virtual network adapter automatically created for each `veth` network adapter in each Container. • `vme`*N* is a virtual network adapter automatically created for each network adapter that exists in a virtual machine and operates in the bridged mode.
`Network ID`	ID of the Virtual Network to which the network adapter is connected. Detailed information on Virtual Networks is provided in **Managing Virtual Networks** (p. 133).
`Addresses`	IP address and subnet mask assigned to the network adapter. `dhcp` denotes that the adapter gets its network parameters from a DHCP server.

Creating VLAN Adapters

Parallels Cloud Server allows you to create new VLAN adapters on the Parallels server. You can use these adapters later on to connect your virtual machines and Containers to any of the available Virtual Networks (for more information on Virtual Networks, see **Managing Virtual Networks (p. 133)**). VLAN adapters can be made using the `vznetcfg vlan add` command. To create a new VLAN adapter, you need to specify two parameters.

- VLAN ID—an arbitrary integer number which will uniquely identify the virtual LAN among other VLANs on the server.
- Physical network adapter on the server to which the VLAN is to be bound.

For example, you can execute the following command to make a new VLAN adapter, associate it with a VLAN having the ID of 5 (i.e. with VLAN 5), and attach the VLAN adapter to the `eth0` physical adapter on the server:

```
# vznetcfg vlan add eth0 5
```

To check that the VLAN adapter has been successfully created, execute the following command:

```
# vznetcfg if list
Name        Type     Network ID   Addresses
eth0        nic                   192.168.0.150/22,dhcp
eth0.5      vlan
...
```

VLAN adapters can be easily identified by the `vlan` designation shown in the `Type` column of the command output. As you can see, only one VLAN adapter currently exists on the server. It is assigned the name of `eth0.5`. This name is generated automatically on the basis of the specified VLAN ID and the name of the physical adapter to which the VLAN adapter is tied.

At any time, you can delete the `eth0.5` VLAN adapter and thus destroy VLAN 5 by issuing the following command:

```
# vznetcfg vlan del eth0.5
```

Networking Modes in Parallels Cloud Server

In Parallels Cloud Server, any virtual machine and Container can operate in one of the two networking modes:

- *host-routed*
- *bridged network*

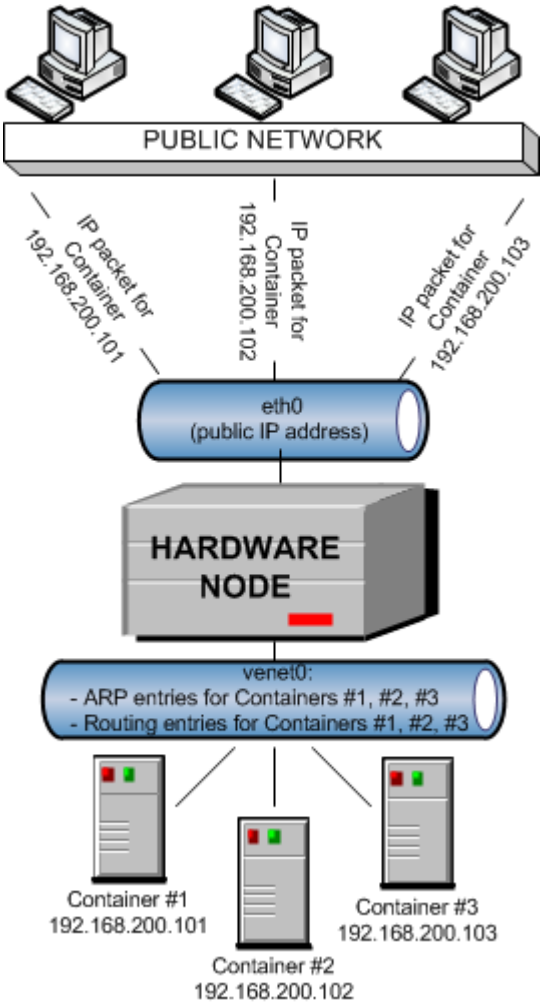
Detailed information on these modes is given in the following sections.

Container Network Modes

This section describes bridged and host-routed network modes for Containers.

Host-Routed Mode for Containers

By default, a new Container starts operating in the host-routed mode. In this mode, the Container uses a special network adapter, `venet0`, to communicate with the server where it resides, with the other Containers on the server, and with computers on external networks. The figure below demonstrates an example network configuration where all Containers are set to work in the host-routed mode.



In this configuration:

- *Container #1, Container #2, Container #3* use the `venet0` adapter as the default gateway to send and receive data to/from other networks. They also use this adapter to exchange the traffic among themselves.
- When *Container #1, Container #2,* and *Container #3* start, the server creates ARP and routing entries for them in its ARP and routing tables. You can view the current ARP and routing entries on a server using the `arp -n` and `route -n` commands. For example:

```
# arp -n
Address                  HWtype  HWaddress           Flags Mask      Iface
10.30.0.4                ether   00:1a:e2:c7:17:c1   C               eth0
10.30.23.162             ether   70:71:bc:42:f6:a0   C               eth0
192.168.200.101          *       *                   MP              eth0
192.168.200.102          *       *                   MP              eth0
192.168.200.103          *       *                   MP              eth0
```

```
# route -n
Kernel IP routing table
Destination       Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.200.101   *               255.255.255.255 UH    1000   0        0 venet0
192.168.200.102   *               255.255.255.255 UH    1000   0        0 venet0
192.168.200.103   *               255.255.255.255 UH    1000   0        0 venet0
10.30.0.0         *               255.255.0.0     U     0      0        0 eth0
default           parallels.com   0.0.0.0         UG    0      0        0 eth0
```

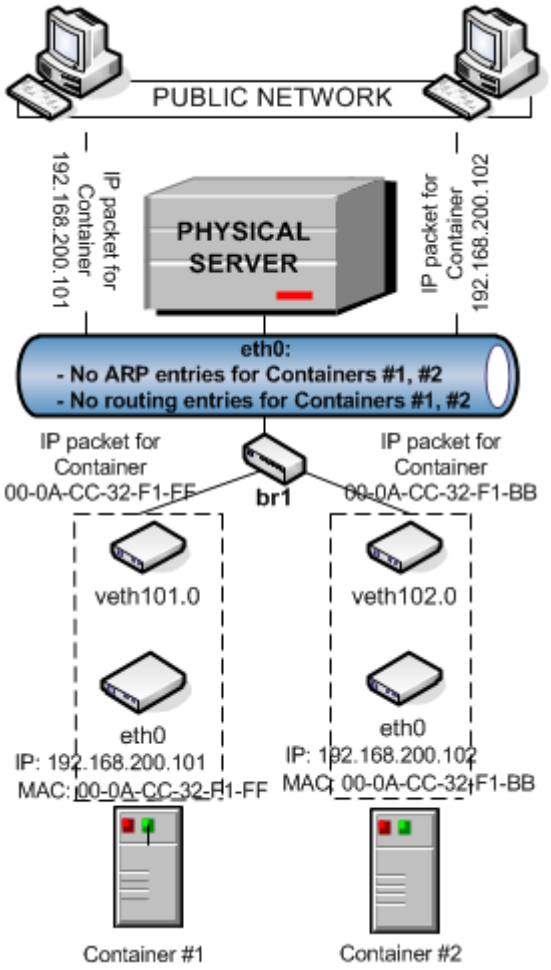
As you can see, the ARP and routing tables contain entries about IP addresses 192.168.200.101, 192.168.200.102, and 192.168.200.103 that belong to Containers #1, #2, and 3#.

Note: The server also stores ARP and routing tables for Containers that have the offline management feature enabled even if they are not running. For more on this feature, see **Configuring Offline Management** (p. 148).

- All Container outgoing network traffic goes to the `venet0` adapter and is forwarded via the `eth0` physical adapter to the destination, according to the routing table of the server.
- All Container incoming network traffic is also processed by the `venet0` adapter. Consider the following situation:
  1. **Computer X** on the local network wants to send a data packet to Container #1 with IP address 192.168.200.101, so it issues an ARP request which computer has this IP address.
  2. The server hosting Container #1 replies with its MAC address.
  3. **Computer X** sends the data packet to the indicated MAC address.
  4. The server receives the packet and transmits it to `venet0` that forwards the packet to Container #1.

Bridged Mode for Containers

The default network adapter of a Container can operate in the host-routed mode only. You can, however, create additional virtual adapters in Containers and make them operate in the bridged network mode. The following figure shows an example network configuration where *Container #1* and *Container #2* are set to work in the bridged mode.



In this configuration:

- Container #1 and Container #2 have separate virtual adapters consisting of two network interfaces:
  - *An ethX interface in the Container (eth0* in the figure). This interface represents a counterpart of a physical network adapter installed on a standalone server. Like any other physical adapter, it has a MAC address, can be assigned one or more IP addresses, included in different networks, and so on.
  - *A vethX interface on the server (veth101.0* and *veth102.0* in the figure). This interface is mostly used to maintain the communication between the server and Ethernet interfaces in Containers.

  Note: To simplify things, virtual adapters operating in the bridged mode are called `veth` adapters, though it is not quite correct from the technical point of view.

  Both interfaces are closely linked to each other, so a data packet entering one interface always comes out from the other one.
- Container #1 and Container #2 keep their own ARP and routing tables that they consult when sending or receiving data.

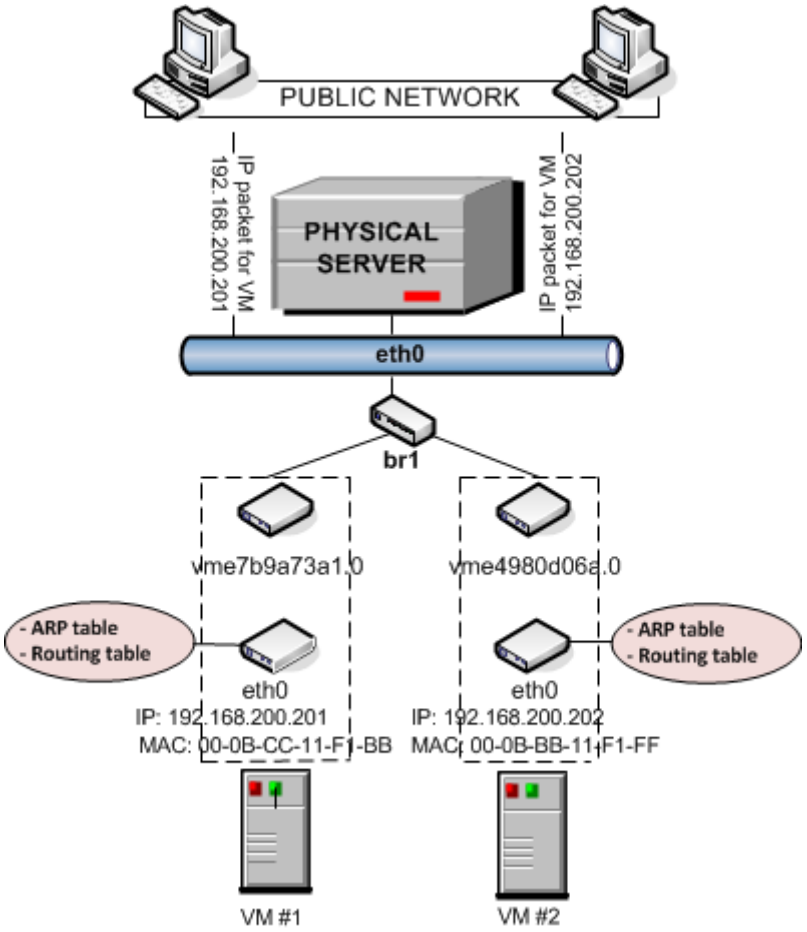
- The `veth` adapters of both Containers are bridged through the bridge `br1` to the physical network adapter `eth0`.
- All Container outgoing traffic comes via the `veth` adapters to the bridge and are then transmitted through the `eth0` physical adapter to the destination, according to the routing tables stored in the Containers.
- All incoming data packets for Container #1 and Container #2 reach the `eth0` physical adapter first and are then sent through the bridge to the `veth` adapter of the destination Container.

Virtual Machine Network Modes

This section describes bridged and host-routed network modes for virtual machines.

Bridged Mode for Virtual Machines

By default, a new virtual machine is created with a network adapter that operates in the bridged mode. The figure below demonstrates an example network configuration where two virtual machines, VM #1 and VM #2, are configured to work in the bridged mode.



In this configuration:

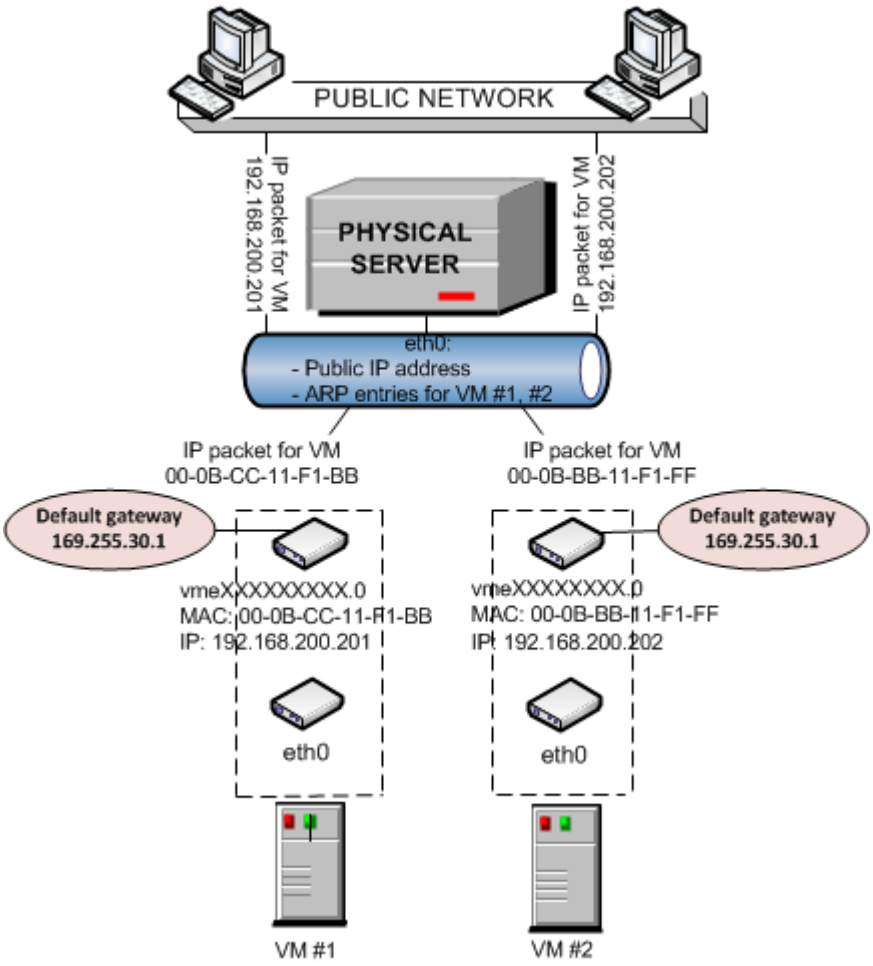
- Each virtual machine has a separate virtual adapter that exposes two interfaces: (1) an `ethX` interface in the virtual machine (`eth0` in the figure) and a `vme` interface on the server (`vme7b9a73a1.0` and `vme4980d06a.0` in the figure). Both interfaces are closely linked to each other, which means that an IP packet entering one interface always comes out from the other one. An `eth` adapter has a MAC address, can be assigned one or more IP addresses, belong to different network environments, and so on.

  Note: To simplify things, virtual adapters operating in the bridged mode are called `vme` adapters, though it is not quite correct from the technical point of view.

- VM #1 and VM #2 keep their own ARP and routing tables that they consult when sending or receiving data.
- The virtual adapters of both virtual machines are bridged through the bridge `br1` to the physical network adapter `eth0`.
- All outgoing data packets are sent from the virtual machines through the bridge and `eth0` physical adapter to the destination, according to their routing tables.
- All incoming data packets for VM #1 and VM #2 reach the `eth0` physical adapter first and are then transmitted through the bridge to the `vme` interface of the destination virtual machine.

Host-Routed Mode for Virtual Machines

The other network mode a virtual machine can work in is the host-routed mode. The figure below demonstrates an example network configuration where three virtual machines, *VM #1*, *VM #2*, and *VM #3*, are set to operate in the host-routed mode.



In this configuration:

- Each virtual machine also has a virtual adapter exposing two interfaces: an `ethX` interface in the virtual machine and a `vmeX` interface on the server.
- Unlike the bridged mode, the ARP entries for VM #1 and VM #2 are stored on the server rather than in the virtual machines themselves. The server creates these ARP entries and saves them to its ARP table when VM #1 and VM #2 start. You can use the `arp -n` command to view the current ARP entries on a server, for example:

```
# arp -n
Address                  HWtype  HWaddress           Flags Mask      Iface
10.30.0.4                ether   00:1a:e2:c7:17:c1   C               eth0
10.30.23.162             ether   70:71:bc:42:f6:a0   C               eth0
192.168.200.201          *       *                   MP              eth0
192.168.200.202          *       *                   MP              eth0
```

- Along with ARP entries, the server also creates routing entries for both virtual machines. So when the server receives a data packet destined for IP address 192.168.200.201, it knows that the packet must be forwarded to the `vmeXXXXXXXX.0` interface of VM #1.
- The server handles all incoming traffic for both virtual machines. Consider the following situation:

1. **Computer X** on the network wants to send a data packet to VM #1 with IP address 192.168.200.201, so it issues an ARP request which computer has this IP address.
2. The server replies with its own MAC address.
3. **Computer X** sends the data packet to the indicated MAC address.
4. The `eth0` physical adapter receives the packet and routes it to the `vme` interface of VM #1.

- All outgoing network traffic sent from VM #1 and VM #2 are routed through the default gateway to the `eth0` adapter on the server. The default gateway for host-routed virtual machines is automatically assigned the IP address of 169.255.30.1. This special IP address is taken from the Automatic Private IP Addressing (APIPA) range and used exclusively to deliver data packets from virtual machines to the server.

Differences Between Host-Routed and Bridged Network Modes

The bridged network mode demonstrates a number of differences as compared to the host-routed one:

- Each `vme` or `veth` virtual adapter has a MAC address assigned to it while a host-routed adapter does not have any. Thanks to this fact:
  - Any virtual machine and Container can see all broadcast and multicast packets received from or sent to the selected network adapter on the server.
  - Using bridged virtual adapters, you can host DHCP or Samba servers in virtual machines and Containers.
- There is no more need to assign all network settings (IP addresses, subnet mask, gateway, and so on) to virtual machines and Containers from the server. All network parameters can be set from inside virtual machines and Containers.
- `veth` and `vme` adapters can be bridged among themselves and with other devices. If several `veth` and `vme` adapters are united into a bridge, this bridge can be used to handle network traffic for the virtual machines and Containers whose `veth` and `vme` adapters are included in the bridge.
- Due to the fact that `veth` and `vme` adapters act as full members on the network (rather than 'hidden' beyond virtual networks adapters on the server), they are more prone to security vulnerabilities: traffic sniffing, IP address collisions, and so on. Therefore, `veth` and `vme` adapters are recommended for use in trusted network environments only.

Configuring Virtual Machines and Containers in Host-Routed Mode

You can configure the following parameters of network adapters that operate in the host-routed mode:

- IP addresses and network masks
- DNS servers
- DNS search domains

Setting IP addresses

The session below shows how to set IP addresses for the `MyVM` virtual machine and Container 101

```
# prlctl set MyVM --device-set net0 --ipadd 10.0.186.100/24
# prlctl set MyVM --device-set net0 --ipadd 1fe80::20c:29ff:fe01:fb07
# prlctl set 101 --ipadd 10.0.186.101/24
# prlctl set 101 --ipadd fe80::20c:29ff:fe01:fb08
```

`net0` in the commands above denotes the network card in the `VM` virtual machine to assign the IP address to. You can view all network cards of a virtual machine using the `prlctl list` *`VM_name`* `-i` command. For Container 101, you do not need to specify the network card name; `prlctl set` automatically performs the operation on the default adapter that always operates in the host-routed mode.

Setting DNS server addresses

To set a DNS server for the `MyVM` virtual machine and Container 101, you can use the following commands.

```
# prlctl set MyVM --device-set net0 --nameserver 192.168.1.165
# prlctl set 101 --nameserver 192.168.1.165
```

Setting DNS search domains

To set a DNS search domain for the `MyVM` virtual machine and Container 101, run these commands:

```
# prlctl set MyVM --device-set net0 --searchdomain 192.168.10.10
# prlctl set 101 --searchdomain 192.168.10.10
```

Notes:

1. You can configure the network settings only of virtual machines that have Parallels Tools installed.

2. Network adapters operating in the routed mode must have at least one static IP address assigned.

3. To assign network masks to Containers operating in the `venet0` networking mode, you must set the `USE_VENET_MASK` parameter in the `/etc/vz/vz.conf` configuration file to `yes`.

4. Containers can have only one network adapter operating in the host-routed mode. This adapter is automatically created when you create a virtual machine.

Switching virtual machine adapters to host-routed mode

By default, a virtual adapter in any newly created virtual machine starts operating in connected to the bridged mode (see the **Connecting Virtual Machines to Virtual Networks** (p. 142) for details). To change the current network mode to host-routed, you can run the following command:

```
# prlctl set VM_Name --device-set Net_ID --type routed
```

For example, to set the `net0` adapter in the `MyVM` virtual machine to operate in the host-routed mode, use this command:

```
# prlctl set MyVM --device-set net0 --type routed
```

Configuring Virtual Machines and Containers in Bridged Mode

This section describes all operations related to configuring virtual machines and Containers that operate in bridged mode.

Managing Virtual Networks

A virtual network acts as a binding interface between a virtual network adapter in a virtual machine and Container and the corresponding network adapter on the Parallels server. Using virtual networks, you can include virtual machines and Containers in different networks. Parallels Cloud Server enables you to manage virtual networks as follows:

- Create a new Virtual Network and remove an existing one.
- Configure the parameters of an existing Virtual Network.
- List the existing Virtual Networks.
- Delete a Virtual Network that you do not need any more.

These operations are described in the following subsections in detail.

Creating a Virtual Network

Virtual networks serve as binding interfaces between the virtual network adapters in virtual machines and Containers and the physical, VLAN, and virtual network adapters on the Parallels server. Using virtual networks, you can connect virtual machines and Containers to different networks.

By default, Parallels Cloud Server creates the following virtual networks on the server:

- *Bridged*. This virtual network is connected to one of the physical adapters on the Parallels server (as a rule, `eth0`) and provides virtual machines and Containers included in this virtual network with access to the network behind this physical adapter.
- *Host-only*. This virtual network is connected to a special virtual adapter on the server and allows the virtual machines and Containers joined to this virtual network to access only the server and the other virtual machines and Containers on this network.

You can create your own virtual networks using the `prlsrvctl` or `vznetcfg` utility. For example, to make a new virtual network with the name of `network1`, you can rung one of the following commands:

```
# vznetcfg net new network1
```

or

```
# prlsrvctl net add network1
```

By default, both commands create host-only virtual networks. However, you can change their types using the `prlsrvctl` utility; see **Configuring Virtual Network Parameters** (p. 135) for details.

In the current version of Parallels Cloud Server, you can create

- Up to 16 host-only virtual networks.
- One or more bridged virtual networks. The number of virtual networks depends on the number of physical and VLAN adapters available on the Parallels server. One virtual network can be connected to only one physical or VLAN adapter.

Viewing Bridges

A virtual network is associated with a bridge that is automatically made on the Parallels server when you create the virtual network and serves as the basis for virtual network operation. To find out what bridge is associated with what virtual network, you can run the following command:

```
# vznetcfg if list
Name Type   Network ID Addresses
eth0 nic    Bridged    10.31.252.116/16,fe80::2a9:40ff:fe0f:b6f2/64,dhcp
br1  bridge Bridged    fe80::200:ff:fe00:0/64
br0  bridge Host-Only  10.37.130.2/24,fdb2:2c26:f4e4::1/64,fe80::200:ff:fe00:0/64
```

The bridges existing on the Parallels server are listed in the `Name` column and can be easily identified by the `br` prefix.

Note: Detailed information on the `vznetcfg` and `prlsrvctl` utilities is provided in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Configuring Virtual Network Parameters

Parallels Cloud Server allows you to configure the following parameters for a virtual network:

- the name assigned to the virtual network
- the networking mode in which the virtual network is operating
- the description of the virtual network

All these operations can be performed using the `prlsrvctl` utility. Let us assume that you want to configure the `pcsnet1` virtual network. This virtual network is currently configured as a host-only network and has the following description set: `This is a host-only virtual network`. To change these parameters, you can execute the following command:

```
# prlsrvctl net set pcsnet1 -n network1 -t bridged --ifname eth1 -d "This is now a
bridged virtual network"
```

This command configured the `pcsnet1` virtual network as follows:

1. Changes the virtual network name to `network1`.
2. Changes the virtual network type to bridged.
3. Changes the virtual network description to the following: `This is now a bridged virtual network`.

For more information on the `prlsrvctl` utility, refer to the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Listing Virtual Networks

To list the virtual networks existing on the Parallels server, you can use either the `vznetcfg` or `prlsrvctl` utility.

Listing virtual networks with vznetcfg

To list the virtual networks on your server using the `vznetcfg` utility, execute the following command:

```
# vznetcfg net list
Network ID        Status      Master Interface  Slave Interfaces
Host-Only         active
Bridged           active      eth0
pcsnet1           active      eth1
```

In the example above, three virtual networks — `pcsnet1` and two default virtual networks — exist on the Parallels server. The information on these virtual networks is presented in the table with the following columns:

Column	Description
Network ID	The ID assigned to the virtual network.
Status	Indicates the status of the virtual network. It can be one of the following: • active: the virtual network is up and running. • configured: the information on the virtual network is present in the /etc/vz/vznet.conf file on the server, but the bridge to which the virtual network is bound is down or does not exist. **Note:** Detailed information on the vznet.conf file is given in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.
Master Interface	Displays the adapter on the server connected to the virtual network, if any.
Slave Interfaces	Lists the adapters in virtual machines and Containers joined to the virtual network, if any.

Listing virtual networks with prlsrvctl

You can also use the prlsrvctl utility to list the virtual networks existing on your server. To do this, run the following command:

```
# prlsrvctl net list
Network ID      Type         Bound To
Host-Only       host-only
Bridged         bridged      eth0
pcsnet1         bridged      eth1
```

This utility displays the following information on virtual networks:

Column	Description
Network ID	The name assigned to the virtual network.
Type	The networking mode set for the virtual network.
Bound To	The adapter on the Parallels server connected to the virtual networks, if any.

Connecting Virtual Networks to Adapters

By connecting an adapter on the physical server to a virtual network, you can join all virtual machines and Containers included in the virtual network to the network to which the corresponding adapter is connected.

Let us assume the following:

- The eth1 physical adapter and the pcsnet1 virtual network exist on the Parallels server. For information on creating virtual networks, see **Creating a Virtual Network** (p. 134).
- The eth1 physical adapter is connected to the local network.
- The MyVM virtual machine is connected to the pcsnet1 virtual network. Detailed information on

joining virtual machines and Containers to virtual networks is given in **Connecting Containers to Virtual Networks** (p. 139) and **Connecting Virtual Machines to Virtual Networks** (p. 142).

To connect the `eth1` adapter to the `pcsnet1` virtual network and thus to join the `MyVM` virtual machine to the network behind `eth1`, run this command on the server:

```
# vznetcfg net addif pcsnet1 eth1
```

To check that the `eth1` physical adapter has been successfully added to the `pcsnet1` virtual network, you can execute the following command:

```
# vznetcfg if list
Name      Type    Network ID    Addresses
eth1      nic     pcsnet1      10.31.252.116/16,fe80::2a9:40ff:fe0f:b6f2/64,dhcp
...
```

As you can see, the `eth1` adapter is now joined to the `pcsnet1` virtual network. That means that the `MyVM` virtual machine whose virtual network adapter is connected to `pcsnet1` can access the local network behind `eth1`.

At any time, you can disconnect the `eth1` physical adapter from the `pcsnet1` virtual network (and thus detach the `MyVM` virtual machine from the local network) by running the following command:

```
# vznetcfg net delif eth1
```

Deleting Virtual Networks

At any time, you can remove a virtual network that you do not need any more from the physical server. To do this, you can use both the `vznetcfg` and `prlsrvctl` utilities. For example, you can delete the `pcsnet1` virtual network by running one of the following commands:

```
# vznetcfg net del pcsnet1
```

or

```
# prlsrvctl net del pcsnet1
```

To check that `pcsnet1` has been successfully removed, execute one of these commands:

```
# vznetcfg net list
Network ID        Status      Master Interface  Slave Interfaces
Host-Only         active
Bridged           active      eth0
```

or

```
# prlsrvctl net list
Network ID     Type        Bound To
Host-Only      host-only
Bridged        bridged     eth0
```

Note: Detailed information on the `vznetcfg` and `prlsrvctl` utilities is provided in the *Parallels Cloud Server 6.0 Command Line Reference Guide* and corresponding Linux manual pages.

Managing Adapters in Containers

Parallels Cloud Server provides you with ample opportunities of configuring `veth` virtual network adapters in Containers and including them in different network environments. This section shows you the way to perform the following operations:

- Create new virtual network adapters in Containers and delete existing ones.
- Configure the parameters of an existing virtual network adapter.
- Join Container virtual network adapters to virtual networks.

All these operations are described in the following subsections in detail.

Creating and Deleting veth Network Adapters

By default, any Container on the Parallels server starts functioning in the `venet0` mode right after its creation. However, at any time you can create additional virtual adapters for Containers and set them to work in the bridged mode. You can do this using the `--netif_add` option of the `prlctl set` command.

Let us assume that you wish to create a new virtual adapter with the name of `eth1` in Container 101 and make it function in the bridged mode. To do this, run the following command :

```
# prlctl set 101 --netif_add eth1
```

The settings of the newly created virtual adapter are saved as the value of the `NETIF` parameter in the configuration file of Container 101 (`/etc/vz/conf/101.conf`). So, you can use the following command to display the parameters assigned to the `veth` network adapter in Container 101:

```
# grep NETIF /etc/vz/conf/101.conf
NETIF="ifname=eth1,mac=00:10:41:F0:AA:B6,host_mac=00:18:51:A0:8A:D7"
```

As you can see, the parameters set for the `veth` virtual network adapter during its creation are the following:

- `ifname`: the name set for the `veth` Ethernet interface in Container 101. You specified this name when creating the Container virtual network adapter. Usually, names of Ethernet interfaces in Containers are set in the form of `eth`*`Ad_N`* where *`Ad_N`* denotes the index number of the created adapter (for example, `eth0` or `eth1`). However, you can choose any other name you like and specify it during the virtual adapter creation.
- `mac`: the MAC address assigned to the `veth` Ethernet interface in Container 101.
- `host_mac`: the MAC address assigned to the `veth` Ethernet interface on the Parallels server.

`ifname` is the only mandatory parameter that you need to specify when creating a Container virtual network adapter. All the other parameters are optional and generated by Parallels Cloud Server automatically, if not indicated.

At any time, you can remove the `veth` virtual network adapter from Container 101 by executing the following command:

```
# prlctl set 101 --netif_del eth1
```

Configuring veth Adapter Parameters

While functioning in the bridged mode, each Container virtual network adapter appears as a full participant on the network to which it is connected and needs to have its own identity on this network.

Fist of all, to start functioning on a TCP/IP network, a `veth` virtual adapter should be assigned an IP address. This can be done as follows:

```
# prlctl set 101 --ifname eth1 --ipadd 192.168.144.123
```

This command sets an IP address of `192.168.144.123` for the `eth1` adapter in Container 101. If you want to use the Dynamic Host Configuration Protocol (DHCP) to make the `eth1` adapter of Container 101 automatically receive TCP/IP configuration settings, you can issue the following command instead:

```
# prlctl set 101 --ifname eth1 --dhcp yes
```

Any static IP address assigned to the Container virtual network adapter can be removed by executing the following command:

```
# prlctl set 101 --ifname eth1 --ipdel 192.168.144.123
```

You can also delete all IP addresses set for Container 101 at once:

```
# prlctl set 101 --ifname eth1 --ipdel all
```

You may also wish to set the following parameters for a Container network adapter:

- A DNS server that the Container virtual adapter is supposed to use:

```
# prlctl set 101 --ifname eth1 --nameserver 192.168.100.111
```

- A gateway to be used for routing the traffic of the Container virtual adapter:

```
# prlctl set 101 --ifname eth1 --gateway 192.168.111.1
```

Detailed information on all options which can be used with the `prlctl set` command to manage Container adapter parameters is given in the *Parallels Cloud Server 6.0 Command Line Reference Guide* and the `prlctl` manual pages.

Connecting Containers to Virtual Networks

With the implementation of `veth` virtual adapters allowing Containers to function as full participants on the network, it has become possible to include Containers in a wide range of network configurations the most common of which are Ethernet networks and VLANs (virtual local area networks). The process of connecting `veth` virtual network adapters to an Ethernet network or to a VLAN is carried out using certain physical and VLAN adapters, respectively, available on the server and involves completing the following tasks:

1. Creating a virtual network that will act as an intermediary between the `veth` adapters and the physical/VLAN adapter.
2. Connecting the `veth` virtual adapter you want to include in an Ethernet network/VLAN to the virtual network.

3 Joining the virtual network where the veth virtual adapters are included to the corresponding physical/VLAN adapter.

After completing these tasks, the Container virtual network adapters will be able to communicate with any computer on the network (either Ethernet or VLAN) where they are included and have no direct access to the computers joined to other networks.

For details on creating new virtual networks and joining physical and VLAN adapters to them, see **Creating a Virtual Network** (p. 134) and **Connecting Virtual Networks to Adapters** (p. 136), respectively. In the example below we assume the following:

- The eth0 physical adapter and the pcsnet1 virtual network exist on the server.
- The eth0 physical adapter is connected to the local Ethernet network and to the pcsnet1 virtual network.
- You want to connect Container 101 and Container 102 to the local Ethernet network.

To join Container 101 and 102 to the local Ethernet network behind the eth0 adapter, you need connect these Containers to the pcsnet1 virtual network. To do this:

1 Find out the name of the veth Ethernet interfaces in Container 101 and 102:

```
# vzlist -a -o ctid,ifname
  CTID  IFNAME
   101  eth1
   102  eth0
   103  -
```

The command output shows that the veth Ethernet interfaces in Container 101 and 102 have the names of eth1 and eth0, respectively.

Note: To add a veth adapter to a virtual network, you must use the name of its Ethernet interface in the Container.

2 Join the veth adapters to the pcsnet1 virtual network:

- Add the veth adapter of Container 101 to the virtual network:

```
# prlctl set 101 --ifname eth1 --network pcsnet1
```

- Add the veth adapter of Container 102 to the virtual network:

```
# prlctl set 102 --ifname eth0 --network pcsnet1
```

After completing these tasks, Container 101 and Container 102 will be able to access any of the servers in the network where the eth0 physical adapter is connected.

At any time, you can disconnect the veth virtual network adapters of Container 101 and 102 from the pcsnet1 virtual network by executing the following commands:

- To disconnect the veth adapter of Container 101 from the virtual network:

```
# prlctl set 101 --ifname eth1 --network ""
```

- To disconnect the veth adapter of Container 102 from the virtual network:

```
# prlctl set 102 --ifname eth1 --network ""
```

Managing Adapters in Virtual Machines

This section provides information on how you can manage virtual network adapters in virtual machines. You will learn to do the following:

- Create new virtual network adapters and delete existing ones.
- Configure the parameters of an existing virtual network adapter.
- Join virtual network adapters to virtual networks.

All these operations are described in the following subsections in detail.

Creating and Deleting Virtual Adapters

A virtual machine can have up to 16 virtual network adapters. Each adapter can be connected to a different network. Let us assume that you want to create a new virtual adapter for the `MyVM` virtual machine. To do this, you can execute the following command :

```
# prlctl set MyVM --device-add net
Creating net1 (+) type=host-only iface='default' mac=001C42AF3D69
```

To check that the network adapter (`net1`) has been successfully added to the virtual machine, run this command:

```
# prlctl list --info MyVM
ID: {f3b3d134-f512-324b-b0b1-dbd642f5220b}
Name: Windows XP
...
  net0 (+) type=host-only iface='default' mac=001C42566BCF
  net1 (+) type=host-only iface='default' mac=001C42AF3D69
```

At any time, you can remove the newly created network adapter (`net1`) by executing the following command:

```
# prlctl set MyVM --device-del net1
Remove the net1 device.
```

For the full list of options that can be used when creating a new virtual network adapter, see the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

Configuring Virtual Adapter Parameters

Parallels Cloud Server allows you to configure the following parameters of virtual machine adapters:

Configuring MAC Addresses

If you need for some reason to regenerate the current MAC address of a network adapter, you can use the following command:

```
# prlctl set MyVM --device-set net1 --mac 00:1C:42:2D:74:00
Creating net1 (+) network=Bridged mac=001C422D7400
```

This command sets the MAC address of `00:1C:42:2D:74:00` for the `net1` adapter in the `MyVM` virtual machine. If do not know what MAC address to assign to your virtual adapter, you can make `prlctl set` automatically generate a new MAC address. To do this, run the following command:

```
# prlctl set MyVM --device-set net1 --mac auto
Creating net1 (+) network=Bridged mac=001C42C84F3E
```

Configuring IP Parameters

As any other standalone server, each virtual machine must have a number of TCP/IP settings configured in the proper way to successfully operate on the network. These settings include:

- IP address
- default gateway
- DNS server

Usually, you define all these settings when you create the virtual machine. However, if you have not yet set any of the settings or want to modify any of them, you can use the `prlctl set` command. For example, you can execute the following command to assign the IP address of `192.129.129.20` to the `net1` adapter in the `MyVM` virtual machine, set the default gateway to `192.129.129.1` and the DNS server to `192.192.192.10`:

```
# prlctl set MyVM --device-set net1 --ipadd 192.129.129.20 --gw 192.129.129.1 --
nameserver 192.192.192.10
```

Along with a static assignment of network parameters to a virtual adapter, you can make the adapter receive its TCP/IP settings automatically using the Dynamic Host Configuration Protocol (DHCP). For example, you can run this command to make the `net1` adapter in the `MyVM` virtual machine get its IP settings through DHCP:

```
# prlctl set MyVM --device-set net1 --dhcp yes
Creating net1 (+) network=Bridged mac=001C42C84F3E
Enable automatic reconfiguration for this network adapter.
```

Notes:

1. You can configure the network parameters only of those virtual machines that have Parallels Tools installed.

2. Detailed information on all options which can be used with the `prlctl set` command to manage virtual machine adapter parameters is given in the *Parallels Cloud Server 6.0 Command Line Reference Guide* and the `prlctl` manual pages.

Connecting Virtual Machines to Virtual Networks

In Parallels Cloud Server, you can connect virtual machines to virtual networks of the following types:

- *Bridged*. This type of virtual network allows the virtual machine to use one of the physical server's network adapters, which makes it appear as a separate computer on the network the corresponding adapter belongs to.

- *Host-only*. This type of virtual network allows the virtual machine to access only the Parallels server and the virtual machines joined to this network.

By default, any newly created adapter is connected to the `Bridged` network. To join a virtual machine to another network, use the `prlctl set` command. For example, the following session demonstrates how you can connect the `net0` adapter of the `MyVM` virtual machine to the `pcsnet1` virtual network.

Before connecting the `MyVM` virtual machine to the `pcsnet1` virtual network, you may wish to check the network adapter associated with this virtual network. You can do it, for example, using the following command:

```
# prlsrvctl net list
Network ID      Type         Bound To
Host-Only       host-only
Bridged         bridged      eth0
pcsnet1         bridged      eth1
```

From the command output, you can see that the `pcsnet1` virtual network is attached to the `eth1` physical adapter on the Parallels server. That means that, after connecting the `MyVM` virtual machine to the `pcsnet1` virtual network, the virtual machine will be able to access all computers on the network where the `eth1` adapter is connected.

Now you can run the following command to join the `net1` adapter of the `MyVM` virtual machine to the `pcsnet1` virtual network:

```
# prlctl set MyVM --device-set net0 --network pcsnet1
Creating net0 (+) network=pcsnet1 mac=001C422D7493
```

Managing Private Networks

This section describes how to manage private networks and subnetworks in Parallels Cloud Server 6.0.

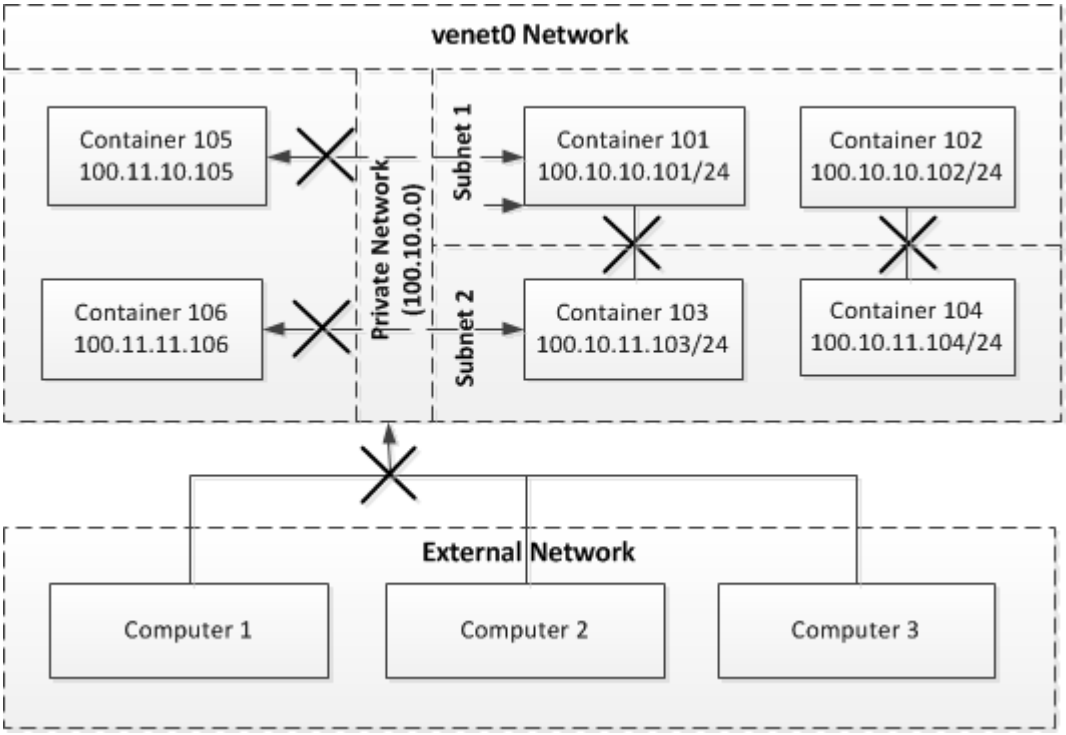
Learning Private Networks

By default, all virtual machines and Containers on the physical server operating in the host-routed mode can access each other even if you connect them to different subnets. For example, if Container 101 has the IP address of 100.10.10.101 and the `MyVM` virtual machine has the IP address of 100.10.11.102 and you set the subnet mask for them to 255.255.255.0, the virtual machine and Container will be able to communicate with each other, though they technically belong to different subnets: 100.10.10.0 and 100.10.11.0.

Note: You can also include virtual machines and Containers operating in the bridged mode in private networks. For information on how you can do this, see **Setting Up Private Networks** (p. 146).

In Parallels Cloud Server 6.0, you can create the so-called *private networks*. Within these private networks, you can make subnets and connect virtual machines and Containers to these subnets so

that the virtual machines and Containers from one subnet will not be able to access virtual machines and Containers from other subnets, virtual machines and Containers outside the private network, and computers on external networks. The following figure demonstrates a system containing a private network:



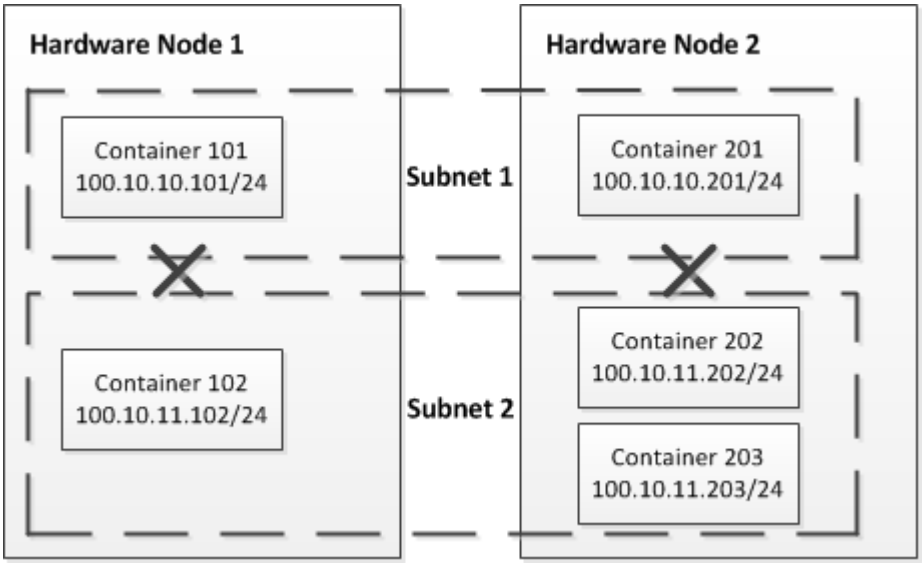
In this example, the network is configured as follows:

- A private network (*Private Network*) is created within the physical server network (*Host Network*).
- The private network contains two private subnets: *Subnet 1* and *Subnet 2*.
- Container 101 and VM1 are connected to Subnet 1, and Container 103 and VM2 are joined to Subnet 2.
- Container 105 and VM3 do not belong to the private network.
- The physical server network is connected to an external network (*External Network*) that contains computers *Computer 1*, *Computer 2*, and *Computer 3*.

In this network, Container 101 and VM1 can access each other, but cannot connect to Container 103, Container 105, VM2, and VM3. Container 103 and VM2, in turn, can also access each other, but cannot connect to Container 101, Container 105, VM1, and VM3. None of the virtual machines and Containers in the private network can access computers on the external network.

Network Across Several Nodes

The example above deals with a private network created within one physical server. However, private networks can span virtual machines and Containers on two or more servers. The following figure demonstrates such a network:



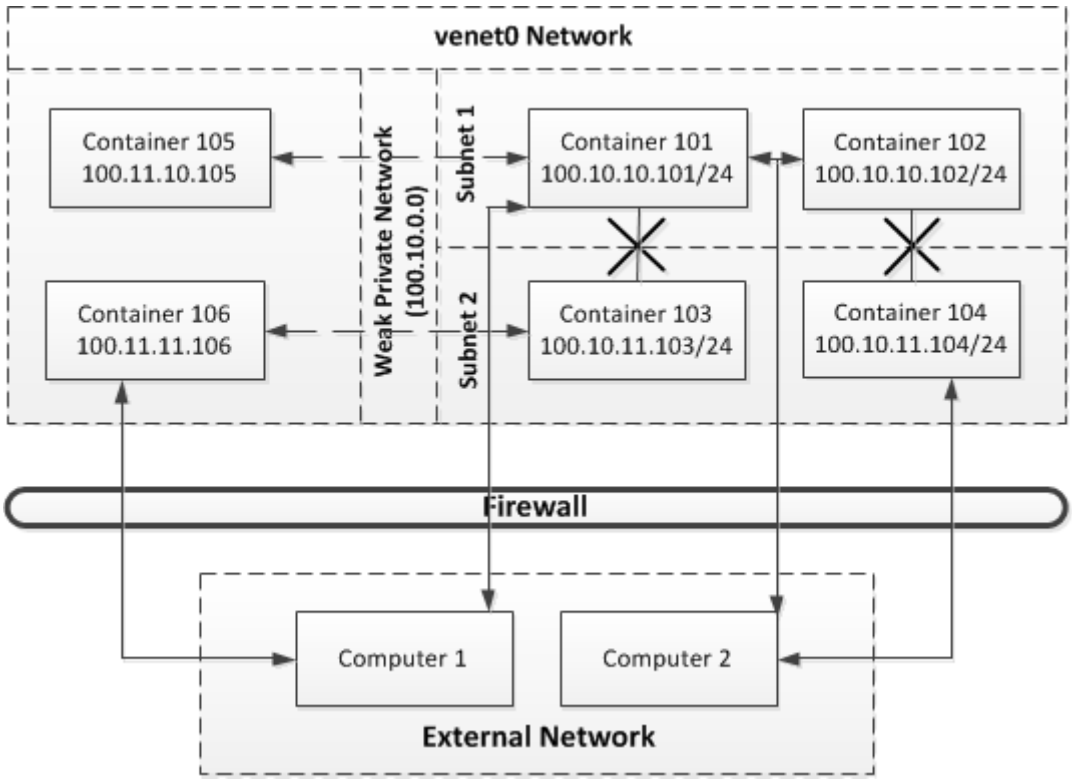
In this figure, the private network also includes two private subnets—Subnet 1 and Subnet 2, but the virtual machines and Containers included in these subnets reside on two physical servers. Container 101 and VM1 are joined to Subnet 1, and Container 102, Container 203, and VM2 are joined to Subnet 2. The virtual machine and Container on Subnet 1 can connect to each other but cannot access the virtual machines and Containers on Subnet 2, and vice versa.

Weak Private Networks

By default, when you create a private network, no virtual machine or Container on this network can access

- virtual machines and Containers that are joined to other subnets in the private network
- virtual machines and Containers that are not part of the private network
- computers that are located on external networks

However, you can configure a private network so that its virtual machines and Containers cannot communicate with virtual machines and Containers on other subnets in the private network but can connect to virtual machines and Containers outside the private network and to computers on external networks. Such private networks are called *weak private networks*. "Weak" in this context means that these networks can be accessed by computers on external networks and are, therefore, more prone to security vulnerabilities and threats. The following picture demonstrates a system with a weak private network:



In this example, the private network on the physical server is divided into two subnets: *Subnet 1* and *Subnet 2*. Container 101 and VM1 are connected to Subnet 1, and Container 103 and VM2 are joined to Subnet 2. Container 105 and VM3 do not belong to the private network. Container 101 and VM1 can access each other, but cannot connect to Container 103 and VM2. Container 103 and VM2, in turn, can also access each other, but cannot connect to Container 101 and VM1.

All virtual machines and Containers in the private network can communicate with Container 105 and VM3 and, as they have public IP addresses assigned, can also access computers on other networks (for example, the computers *Computer 1* and *Computer 2* on the external network *External Network*). To protect the virtual machines and Containers from possible security vulnerabilities and threats, the firewall is configured on the physical server, blocking unauthorized access to the virtual machines and Containers.

Setting Up Private Networks

Before you start setting up a private network, you need to decide on the following:

1. The name to assign to the private network.
2. The range of IP addresses to allocate to the private network.
3. The number of subnets and hosts in the private network.

Let us assume that you want to create two private networks with the following parameters:

- The first network has the name `privnet1` and includes the IP addresses from 10.10.0.0

through 10.10.255.255.

- The second network has the name `privnet2` and includes the IP addresses from 10.11.0.0 through 10.11.255.255.

To create these two networks, you can run the following commands on the physical server:

```
# prlsrvctl privnet add privnet1 --ipadd 10.10.0.0/16
# prlsrvctl privnet add privnet2 --ipadd 10.11.0.0/16
```

Now if you assign to one virtual machine or Container an IP address from the range 10.10.0.0 through 10.10.255.255 and to another virtual machine or Container from the range 10.11.0.0 through 10.11.255.255, they will not be able to access each other because they will belong to different private networks.

If you want to create a private network for several virtual machines and Containers only, you can omit the network mask and specify only the required IP addresses:

```
# prlsrvctl privnet add privnet3 --ipadd 10.12.0.101
# prlsrvctl privnet add privnet3 --ipadd 10.12.0.102
```

The commands above create the private network `privnet3` and include only two IP addresses in this network: 10.12.0.101 and 10.12.0.102. You can then assign these IP address to the two of your virtual machines and Containers, thus isolating their network traffic from that of other virtual machines and Containers and external computers.

Enabling Private Networks for virtual machines and Containers Operating in Bridged Mode

By default, you can include in private networks only virtual machines and Containers operating in the host-routed mode. If you want to connect to a private network some of virtual machines and Containers operating in the bridged mode, you need first to enable the private network support on the physical server. To do this, change the value in the `/proc/sys/net/vzpriv_handle_bridge` file from 0 to 1:

```
# echo 1 > /proc/sys/net/vzpriv_handle_bridge
```

Note: Enabling the support for private networks may affect the network performance of virtual machines and Containers that operate in the bridged mode and are assigned IPv4 addresses.

Creating Weak Private Networks

In a weak private network, any virtual machine or Container on the network can communicate with the other virtual machines and Containers in the same subnet, virtual machines and Containers outside the private network, and computers on external networks. To create a weak private network, you can run the following command:

```
# prlsrvctl privnet set privnet_name --ipadd '*'
```

where *`privnet_name`* is the name of the private network you want to configure as a weak one. For example, to configure the `privnet1` private network as a weak one, execute the following command:

```
# prlsrvctl privnet set privnet1 --ipadd '*'
```

To make sure that privnet1 is now a weak network, check the contents of the /proc/vz/privnet/sparse file:

```
# cat /proc/vz/privnet/sparse
1: * 10.10.0.0/16
2:   10.11.0.0/16
```

The asterisk before 10.10.0.0/16 denotes that privnet1 is now acting as a weak network. To revert the changes made to the privnet1 file, run this command:

```
# prlsrvctl privnet set privnet1 --ipdel '*'
```

Connecting virtual machines and Containers to Private Subnets

Once you set up a private network, you can connect Containers to different subnets within this network. Assuming that you followed the instructions above, you now have two private networks. The privnet1 network includes the IP addresses from 10.10.0.0 through 10.10.255.255, and the privnet2 network contains the IP addresses from 10.11.0.0 through 10.11.255.255. Let us join Container 101 to privnet1 and the MyVM virtual machine to privnet2. To do this:

1 Assign IP address 10.10.10.101 to Container 101:

```
# prlctl set 101 --ipadd 10.10.10.101
```

2 Assign the IP address of 10.11.10.101 to the MyVM virtual machine:

```
# prlctl set MyVM --device-set net0 --ipadd 10.11.10.101
```

Now Container 101 and the MyVM virtual machine belong to different subnets and cannot access each other.

Removing Private Networks

At any time, you can remove privnet1 and privnet2 by running these commands:

```
# prlsrvctl privnet del privnet1
# prlsrvctl privnet del privnet2
```

Once you execute these commands, all virtual machines and Containers that were included in the privnet1 network should be able to connect to the virtual machines and Containers that were joined to the privnet1 network.

Configuring Offline Management

Using the offline management functionality, you can connect to and manage virtual machines and Containers with the help of your favorite web browser. If offline management is enabled for a virtual machine or Container, you can access it via offline services configured for this virtual machine or Container. When you turn on an offline service, one of Container ports becomes always open no matter what state it has been in, and you can access the virtual machine or Container even if it is stopped.

This section provides the following information about offline management:

- offline management basics (p. 149)
- ways to enable and disable the offline management functionality (p. 152)
- ways to enable and disable offline services (p. 153)
- insight into offline management configuration files (p. 153)

Note: In the current version of Parallels Cloud Server, offline management does not work for Containers operating in the bridged mode.

Understanding Offline Management

Using the offline management functionality, you can connect to and manage virtual machines and Containers with the help of your favorite web browser. If offline management is enabled for a virtual machine or Container, you can access it via offline services configured for this virtual machine or Container. When you turn on an offline service, one of Container ports becomes always open no matter what state it has been in, and you can access the virtual machine or Container even if it is stopped.

The way offline management works slightly differs for

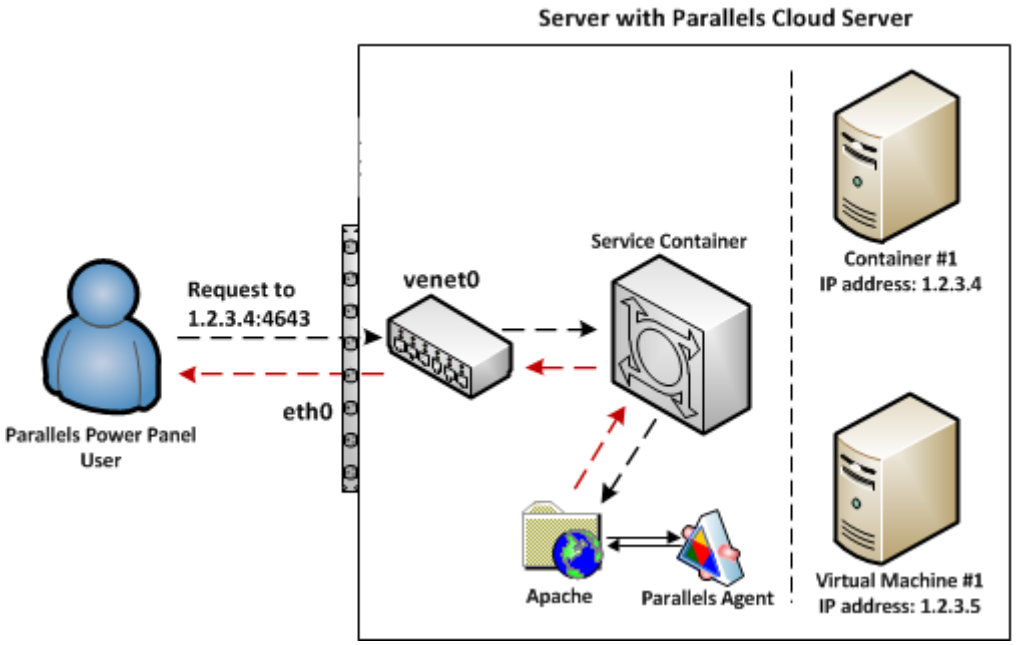
- Containers (both running and stopped) operating in the host-routed mode, running virtual machines operating in the host-routed mode, and stopped virtual machines working in both network modes

  and

- running virtual machines operating in the bridged network mode.

Offline Management for Containers and Stopped Virtual Machines

The following figure shows an example offline management configuration for host-routed Containers (both running and stopped), running host-routed virtual machines, and stopped host-routed and bridged virtual machines:



When offline management is enabled, the server keeps ARP and routing entries for host-routed Containers, running host-routed virtual machines, and all stopped virtual machines. In this case, offline management works as follows:

1. From Parallels Power Panel, a user sends a data packet to port 4643 of IP address 1.2.3.4 that belongs to Container #1 (port 4643 is used by the `vzpp` offline service).

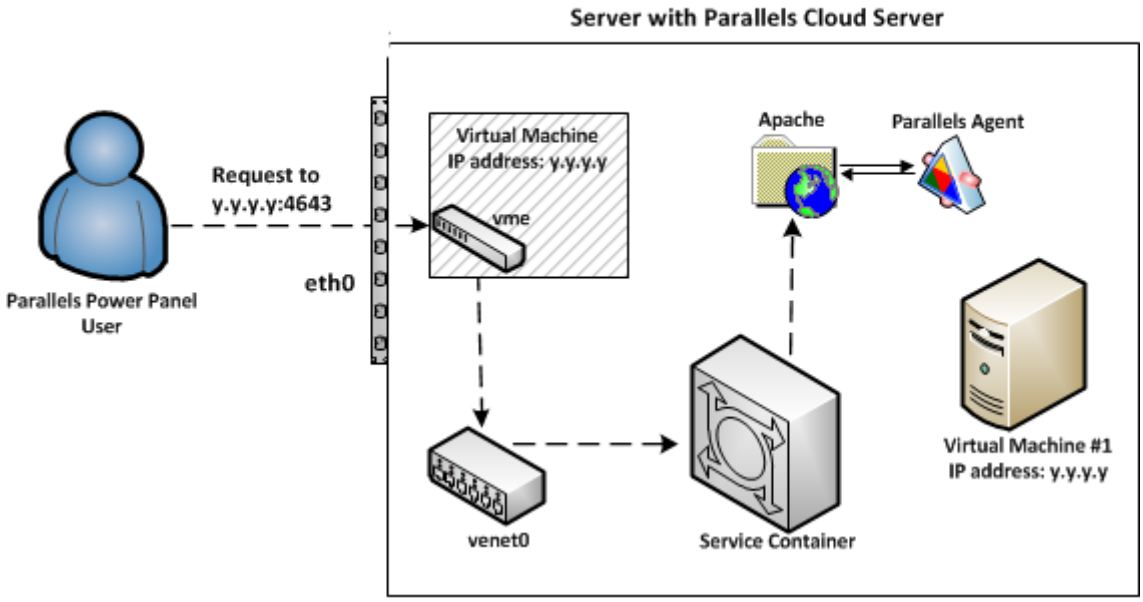
   Note: For the sake of brevity, this description uses Container #1 as an example. The same, however, is true for Virtual Machine #1 as well.

2. The server keeping an ARP entry for Container #1 receives the packet and then sends the packet to the `venet0` adapter.
3. The `venet0` adapter is configured to forward all packets coming to port 4643 to a special Service Container, so it transmits the packet to the Service Container instead of sending it to Container #1.
4. The `init` process running in the Service Container and configured to listen to packets to port 4643 receives the packet and forwards it to the Apache web server running on the server. The Service Container saves the information about the processed packet to a log file in the `/var/opt/pva/pp/sve/fs/root/var/log/pavm` directory on the server
5. The Apache server interacts with the Parallels Agent to process the packet and sends a reply to the Service Container which forwards it to `venet0`.

6 venet0 sends the reply via the eth0 physical adapter back to the Parallels Power Panel user.

Offline Management for Running Bridged Virtual Machines

If a bridged virtual machine is running, the server does not keep any ARP and routing entries for it, and all routing is done via the vme interface of the virtual machine.



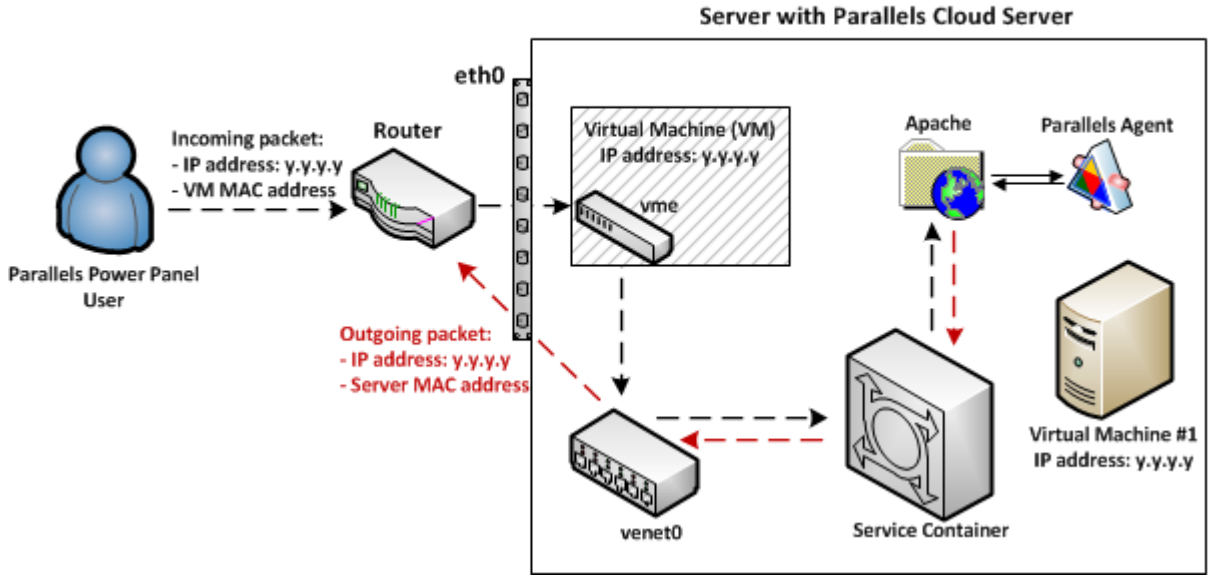
A data packet for a running virtual machine first arrives at its virtual interface (vme in the figure above) and only then is forwarded to venet0. The remaining steps are the same as those described for Containers and stopped virtual machines.

The fact that the packet comes via the vme interface may cause offline management of a running virtual machine to malfunction in some situations. Consider the following situation:

1 A router in your network receives an incoming packet intended for Virtual Machine #1. The packet contains the IP address and MAC address of the destination virtual machine—that is, of Virtual Machine #1.

2 After the packet is processed, it is sent back to the router via venet0. This outgoing packet still contains the IP address of Virtual Machine #1 but the MAC address of the server.

3 When the router sees that the MAC address is different, it might drop the outgoing packet.



To avoid such problems, you can reconfigure the router to accept packets with the same IP address but different MAC addresses. You can also switch virtual machine's network adapter to the host-routed mode or create an additional network adapter and set it to work in the host-routed mode.

Enabling and Disabling Offline Management

By default, offline management is enabled for all newly created virtual machines and Containers. To manage a virtual machine or Container with Parallels Power Panel, just visit its IP address or hostname on port 4643 in a Web browser (e.g., `https://192.168.0.100:4643`).

You can disable offline management

- globally for all virtual machines and Containers on a server, or
- for a particular virtual machine or Container only.

To disable offline management for all virtual machines and Containers on a server, set the `OFFLINE_MANAGEMENT` parameter to `no` in the `/etc/vz/vz.conf` configuration file.

To disable offline management for a particular virtual machine or Container, use the `prlctl set` command with the `--offline_management no` option. For example, to disable offline management for Container 101, run this command:

```
# prlctl set 101 --offline_management no
```

Enabling and Disabling Offline Services

The current version of Parallels Cloud Server supports the following offline services:

- `vzpp` for managing virtual machines and Containers with Parallels Power Panel,
- `vzpp-plesk` for managing virtual machines and Containers with Plesk Control Panel integrated with Parallels Power Panel.

The names of services enabled on a server are listed in the value of the `OFFLINE_SERVICE` parameter in the global configuration file (`/etc/vz/vz.conf`):

```
# cat /etc/vz/vz.conf | grep OFFLINE_SERVICE
OFFLINE_SERVICE="vzpp vzpp-plesk"
```

If a service is enabled in the `/etc/vz/vz.conf` file, you can use it to manage all newly created virtual machines and Containers.

You can disable offline services:

- Globally for all virtual machines and Containers on a server. To do this, remove the desired service from the `/etc/vz/vz.conf` file.
- For individual virtual machines and Containers. To do this, make sure that the offline service is enabled globally in the `/etc/vz/vz.conf` file and use the `prlctl set` command with the `--offline_service` option. For example, to disable the `vzpp` service for Container 101, run this command:

```
# prlctl set 101 --offline_service vzpp
```

Offline Management Configuration Files

The offline management configuration files located in the `/etc/vzredirect.d` directory define various modes of managing virtual machines and Containers via offline management services. Each configuration file describes one offline management mode. In the current Parallels Cloud Server version, two files are accessible:

- `vzpp.conf`. Defines offline management with Parallels Power Panel.
- `vzpp-plesk.conf`. Defines offline management with Plesk Control Panel integrated with Parallels Power Panel.

Each file contains two parameters:

Parameter	Description
`PORT`	Port number you will access a virtual machine or Container on. The default ports are • 4643 for Parallels Power Panel, • 8443 for Plesk Control Panel integrated with Parallels Power Panel.
`DST_VEID`	virtual machine or Container ID to which to redirect requests on `PORT`. By default, all requests are redirected to a special Service Container with the ID of 1.

Using Open vSwitch Bridges

Parallels Cloud Server 6.0 comes with support for Open vSwitch 1.6 (http://openvswitch.org), multi-layer network switches that function as virtual switches, providing virtual machines and containers with network connectivity. By default, support for Open vSwitch is disabled and Linux bridges are used instead. To start using Open vSwitch bridges on your server, complete these steps:

1 Install the `openvswitch` package on the server. This package is available in the remote Parallels Cloud Server repository, so you can use the `yum install` command to install it:

```
# yum install openvswitch
```

2 Configure the `openvswitch` service to start automatically when the server boots:

```
# chkconfig openvswitch on
```

3 Create a configuration file for the Open vSwitch bridge. Name the file `ifcfg-`*`BRIDGE_NAME`*`-`*`NIC_NAME`* and save it to the `/etc/sysconfig/network-scripts` directory. For example, the command below creates the configuration file for the `br-eth0` bridge that will be attached to the `eth0` network card. The easiest way of creating the `ifcfg-br-eth0` configuration file is by copying the `ifcfg-eth0` file with all its contents:

```
# cp /etc/sysconfig/network-scripts/ifcfg-eth0 /etc/sysconfig/network-scripts/ifcfg-br-eth0
```

4 Configure the parameters in the `ifcfg-br-eth0` file:

- Set the value of the `DEVICE` parameter to `br-eth0`.
- Set the value of the `TYPE` parameter to `OVSBridge`.
- Make sure the `SLAVE` parameter is present in the file and set to `yes`.
- Remove the `HWADDR` parameter from the file.

An example of the `ifcfg-br-eth0` file is given below:

```
DEVICE="br-eth0"
TYPE="OVSBridge"
SLAVE="yes"
BOOTPROTO="dhcp"
IPV6INIT="no"
NM_CONTROLLED="yes"
ONBOOT="yes"
DEFROUTE="yes"
PEERDNS="yes"
PEERROUTES="yes"
IPV4_FAILURE_FATAL="yes"
```

5 Configure the `/etc/sysconfig/network-scripts/ifcfg-eth0` file:

- Remove all parameters from the file, except for the following: `DEVICE`, `ONBOOT`, `TYPE`, `HWADDR`, `DEVICETYPE`.
- Set the value of the `TYPE` parameter to `OVSPort`.
- Set the value of the `DEVICETYPE` parameter to `ovs`.

- Add the `OVS_BRIDGE` parameter to the file and set its value to `br-eth0`.

An example of the `ifcfg-eth0` file is given below:

```
DEVICE="eth0"
ONBOOT="yes"
HWADDR="XX:XX:XX:XX:XX:XX"
TYPE="OVSPort"
DEVICETYPE="ovs"
OVS_BRIDGE="br-eth0"
```

6 Restart the server for the changes to take effect.

After restart, you can check that the `br-eth0` bridge has been successfully attached to the `eth0` network card by running this command:

```
# vznetcfg if list
Name          Type      Network ID  Addresses
br-eth0       bridge                10.30.23.246/16,fe80::21c:42ff:fe74:dacf/64,dhcp
veth-eth0.0   veth                  fe80::a465:a3ff:fe0e:57a6/64
veth-eth0.1   veth      Bridged     fe80::c79:efff:fe34:301f/64
br1           bridge    Bridged     fe80::74ef:5dff:fe18:a044/64
br0           bridge    Host-Only   10.37.130.2/24,fdb2:2c26:f4e4::1/64
eth0          nic       Bridged     10.30.23.246/16,fe80::21c:42ff:fe74:dacf/64
```

Open vSwitch Restrictions and Limitations

Listed below are Open vSwitch restrictions and limitations:

- It is impossible to use the `iptables` functionality on physical servers with Open vSwitch bridges enabled.
- Creating and managing private networks with Open vSwitch bridges is not supported. For more information on private networks, see **Managing Private Networks** (p. 143).
- Parallels Automation for Cloud Infrastructure does not support the "Basic Firewall" feature for servers using Open vSwitch bridges.

CHAPTER 7

Managing Licenses

The given chapter provides information on managing Parallels Cloud Server licenses. You will learn how to view the current license status, to install a new license on your server or to update an existing one, to transfer the license from one server to another, etc.

Note: If you want to use the Parallels Cloud Storage functionality, you need to install a separate license in addition to a Parallels Cloud Server license. For detailed information on managing Parallels Cloud Storage licenses, consult the *Parallels Cloud Storage Administrator's Guide*.

In This Chapter

Installing the License 156
Updating the Current License 157
Transferring the License to Another Server 158
Viewing the Current License 159

Installing the License

Depending on the way you have obtained your Parallels Cloud Server license, the process of installing the license slightly differs:

- If you have obtained the license in the form of a product key, you can install it on the server using the `-p` option of the `vzlicload` command. For example, you can execute the following command to install the `XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX` product key:

```
# vzlicload -p XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX
```

  Note: You can also use the `vzlicload` utility to upgrade the license. For example, this may be necessary if your current license does not support using Parallels Virtual Automation for managing Parallels servers.

- If you have obtained the license in the form of an activation code, you can install it on the server using the `-a` option of the `vzlicupdate` command. For example:

```
# vzlicupdate -a XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX
```

  where `XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX` is your activation code. When executed, `vzlicupdate` connects to the Parallels Key Authentication (KA) licensing server and transmits the specified activation code there. The licensing server, in turn, generates a license file, sends it back to the server from where the activation code has been dispatched, and automatically installs it on this server. So, before executing the aforementioned command, make sure that your Parallels server is connected to the Internet.

If you are activating your installation by means of an activation key, you must have an active Internet connection to successfully complete the license installation. Otherwise, you will be presented with the corresponding warning message informing you of the steps you have to take to activate your license. As a rule, these steps are the following:

1. Visiting the http://www.parallels.com/en/support/virtuozzo/activate web page and activating the license manually.
2. Providing the following information on this web page:
   - In the **Product Code** field, specify your license activation code.
   - In the **HWID** field, provide the ID of your server.
   - In the **Enter following digits** field, type the digits displayed next to this field.
3. Clicking the **ACTIVATE LICENSE** button.

If you have entered the correct information on the **Virtuozzo License Activation** page, you will be provided with a link to a license file that you should download to and install on the server. For example, you can run this command to install the obtained license file

```
# vzlicload -f /etc/vzlicense
```

This command will install the license file with the name of `vzlicense` on your server.

Updating the Current License

In Parallels Cloud Server, you can use the `vzlicupdate` utility to update the license currently installed on the Parallels server. When executed, the utility tries to connect to the Parallels Key Authentication (KA) server and to retrieve a new license and install it on the server. To update your license, do the following:

1. Make sure that the Parallels server where you wish to update the license is connected to the Internet.
2. Execute the following command on the server:

```
# vzlicupdate
```

By default, `vzlicupdate` tries to access the KA server having the hostname of `ka.parallels.com`. However, you can explicitly specify what KA server to use using the `--server` option:

```
# vzlicupdate --server ka.server.com
```

In this case, the `vzlicupdate` utility will try to connect to the KA server with the hostname of `ka.server.com`, to get a new license from this server, and to install it on the server where `vzlicupdate` has been executed.

Note: Your physical server must be assigned at least one public IPv4 address for the correct operation of the `vzlicupdate` utility.

Transferring the License to Another Server

Sometimes, you may wish to transfer licenses from one Parallels server (*source server*) to another (*destination server*). For example, this may be the case if the server where the license is installed starts experiencing problems or requires the hardware upgrade.

The procedure of transferring a license from one Parallels server to another depends on the license type and can be one of the following:

Activation with a product key

If you have activated your Parallels Cloud Server installation by means of a product key, you can transfer the installed license from the source to the destination server as follows:

1. Remove the installed license from the source server (e.g., using the `vzlicload -r` *`product_key`* command).
2. Log in to the destination server.
3. Install the product key on the destination server. Detailed information on how to install Parallels Cloud Server licenses is provided in **Installing a License** (p. 156).

Activation with an activation code

If you have activated your Parallels Cloud Server installation by means of an activation code, you can use the `vzlicupdate` utility to move licenses between Parallels servers. For example, to transfer a license that has been installed using the `XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX` activation code, do the following:

1. Ascertain that the source server is shut down, or the license is removed from this server.
2. Make sure that the destination server is up and connected to the Internet.
3. Log in to the destination server (e.g., via `ssh`).
4. Execute the following command on the destination server:

```
# vzlicupdate -t -a XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX
```

When executed, `vzlicupdate` sends the activation code to the Parallels KA server, thus informing the server of its intention to transfer the license to a new Parallels server. The KA server verifies the received code, generates a new license file, sends it back to the destination server, and installs it there.

Note: Your physical server must be assigned at least one public IPv4 address for the correct operation of the `vzlicupdate` utility.

You can check that the license transferal has completed successfully using the `vzlicview` utility. For example:

```
# vzlicview
Show installed licenses...
VZSRV
        status="ACTIVE"
        version=X.X
        serial="XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX"
        expiration="05/01/2012 23:59:59"
         ...
```

Detailed information on the `vzlicview` utility and its output is provided in **Viewing Current License** (p. 159).

Viewing the Current License

The given subsection familiarizes you with the way to view the information on the license installed on your Parallels server.

Viewing the License

In Parallels Cloud Server, you can use the `vzlicview` utility to view the information on the installed license and find out its current status. When executed, this utility processes the license currently installed on the Parallels server and prints the license contents along with its status. A sample output of `vzlicview` is given below:

```
# vzlicview
Searching for installed licenses...
VZSRV
        status="ACTIVE"
        version=X.X
        serial="XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX"
        expiration="12/01/2012 23:59:59"
        graceperiod=86400 (86400)
        key_number="PCS.00000001.0000"
        cpu_total=64 (1)
        ct_total=100 (1)
        platform="Any"
        product="PCS"
        nr_vms=10 (2)
        architecture="Any"
        ha_allowed=1
        rku_allowed=0
```

The command output shows the full information about the license. The main license parameters are listed in the following table:

Column	Description
`status`	The license status. The information on all possible license statuses is provided in **License Statuses** (p. 161).
`version`	The version of Parallels Cloud Server for which the license was issued.
`serial`	The license serial number.
`expiration`	The license expiration date, if it is time-limited.

graceperiod	The period, in seconds, during which Parallels Cloud Server continues functioning if • the license has expired • the number of running virtual machines and Containers exceeds the limit defined by the license
key_number	The number under which the license is registered on the Parallels Key Authentication server.
cpu_total	The total number of CPUs you are allowed to install on the Parallels server.
ct_total	The total number of Containers you are allowed to simultaneously run on the Parallels server.
platform	The operating system with which the license is compatible.
product	The product name for which the license has been issued.
nr_vms	The number of virtual machines you are allowed to simultaneously run on the Parallels server.
architecture	The system architecture with which the license is compatible.
concerto	If this field is present, the license supports the ability to use the Plesk application in Containers.
rku_allowed	Denotes whether the rebootless upgrades feature is enabled (1) or disabled (0) in the license.
ha_allowed	Denotes whether the Parallels Cloud Storage functionality is enabled (1) or disabled (0) in the license.

Licenses with a combined limit on virtual machines and Containers

Some licenses shipped with Parallels Cloud Server 6.0 define a combined limit on the number of virtual machines and Containers you are allowed to simultaneously run on the Parallels server rather than set limits separately for virtual machines and Containers. In this case, the license output is as follows:

```
# vzlicview
Searching for installed licenses...
VZSRV
        status="ACTIVE"
        version=X.X
        serial="XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX"
        expiration="12/01/2013 23:59:59"
        graceperiod=86400 (86400)
        key_number="PCS.00000001.0000"
        cpu_total=64 (1)
        ct_total=100 (1)
        platform="Any"
        product="PCS"
        nr_vms="combined" (2)
        servers_total=100
        architecture="Any"
```

As you can see, the output now contains one more parameter—`servers_total`. This parameter defines the total number of virtual machines and Containers you can simultaneously run on the

Parallels server. For example, according to the license above, you can run 100 Containers, or 100 virtual machines, or 50 Containers and 50 virtual machines on the server at the same time.

License Statuses

When viewing information on your license, pay special attention to the license status that can be one of the following:

Status	Description
ACTIVE	The license installed on the server is valid and active.
VALID	The license the utility parses is valid and can be installed on the server.
EXPIRED	The license has expired and, therefore, could not be installed on the server.
GRACED	The license has been successfully installed on the server; however, it is currently on the grace period because • the license has expired • the number of running virtual machines and Containers exceeds the limit defined by the license
INVALID	The license is invalid (for example, because of the server architecture mismatch) or corrupted.

CHAPTER 8

Keeping Your System Up To Date

This chapter explains the ways to keep your Parallels server up to date. The components you need to take care of are the following:

- Parallels Cloud Server software (p. 162)
- virtual machines (p. 164) and Containers (p. 165) hosted on the server

In This Chapter

Updating Parallels Cloud Server 162
Updating Software In Virtual Machines 164
Updating Containers 165

Updating Parallels Cloud Server

Parallels Cloud Server allows quick and easy updates with the `yum` utility standard for RPM-compatible Linux operating systems. The main components you may need to update are the following:

- Utilities and libraries (p. 163)
- Kernel (p. 163)
- EZ templates (p. 164)

Note: The `vzup2date` utility is no longer supported in Parallels Cloud Server.

Rebootless Updates

In earlier versions of Parallels Server Bare Metal, applying updates to your system often required rebooting the server. During the reboot process, all virtual machines and Containers running on the server were shut down and started again. If the number of running virtual machines and Containers was high enough, stopping and starting each of them could result in significant downtime.

Starting with version 6.0, Parallels Cloud Server supports rebootless updates. During a rebootless update, all running virtual machines and Containers are suspended in the server's memory and then resumed rather than shut down and started again. This greatly reduces their downtime and virtually eliminates the service outage or interruption for end users.

To perform a rebootless update:

1 Download and install the new Parallels Cloud Server kernel using `yum`. For detailed information on updating kernels, see **Updating Kernel** (p. 163).

2 Run the `vzreboot` command:

```
# vzreboot
```

This command checks the GRUB configuration file to determine the default kernel, loads this kernel into memory, and then boots the new kernel. You can also specify the kernel version manually, for example:

```
# vzreboot 2.6.32-042stab054.2
```

In this case, the command will load and then boot the specified kernel instead of the one set as default in the GRUB configuration file.

Updating All Components

The easiest way to update all components of the Parallels Cloud Server software is to simply run the `yum update` command. When executed, this command tells the `yum` utility to do the following:

1 Access the Parallels remote repositories.

2 Check for available updates for the Parallels Cloud Server kernel, utilities, libraries, and EZ templates.

3 Install the found updates on your system.

Note that the `yum` utility can only update the packages that are already installed on the server. So if a package is not available on your system, you first need to install the package using the `yum install` command.

Updating Kernel

Updating the Parallels Cloud Server kernel includes updating the following packages:

- `vzkernel`
- `vzkernel-devel`
- `vzmodules`
- `vzmodules-devel`

To update these, run the `yum update` command as follows, separating package names by white spaces:

```
# yum update vzkernel vzkernel-devel vzmodules vzmodules-devel
...
Installed:
  vzkernel.x86_64 0:2.6.32-042stab049.5  vzkernel-devel.x86_64 0:2.6.32-042stab049.5
vzmodules.x86_64 0:2.6.32-042stab049.5  vzmodules-devel.x86_64 0:2.6.32-042stab049.5
```

```
Complete!
```

Updating EZ Templates

You can update an EZ template like any other RPM package using the `yum update` command. For example:

```
# yum update centos-6-x86-ez
...
Updated:
  centos-6-x86-ez.noarch 0:4.7.0-1
Complete!
```

Notes:

1. Updating an OS EZ template requires that you append `ez` to its name.

2. You can also use the `vzpkg update template` command to update EZ templates. For detailed information on using this command, consult the *Parallels Cloud Server 6.0 Templates Management Guide*.

Checking for Updates

Before updating any packages, you may want to see which can be updated and to what version. You can do that with the `yum check-update` command. For example:

```
# yum check-update
...
vzkernel.x86_64                 2.6.32-042stab049.5           pcs-base
vzkernel-devel.x86_64           2.6.32-042stab049.5           pcs-base
vzmodules.x86_64                2.6.32-042stab049.5           pcs-base
vzmodules-devel.x86_64          2.6.32-042stab049.5           pcs-base
```

Performing More Actions

The `yum` command allows you to do more than just check for and install updates. Some of the other useful options that may help you in updating Parallels Cloud Server are `search`, `list`, `info`, `deplist`, `provide`. For more information on these and other options, see the `yum` Linux manual page.

Updating Software In Virtual Machines

To keep software in your virtual machines up to date, you can use the same means you would use on standalone computers running the corresponding operating systems:

- In Linux-based virtual machines, you can use the native Linux updaters (`up2date`, `yum`, or `yast`).
- In Windows-based virtual machines, you can use the native Windows updaters (e.g. the

Windows Update tool).

You should regularly run these updaters to ensure that your system has the latest updates and fixes (including security patches) installed. For more information on native updaters, refer to the documentation shipped with your operating system.

Updating Containers

Parallels Cloud Server provides two facilities to keep your Containers up to date. These facilities include:

- Updating EZ templates software packages inside a particular Container by means of the `vzpkg` utility. Using this facility, you can keep any of the Containers existing on your Parallels server up to date.
- Updating caches of the OS EZ templates installed on the Parallels server. This facility allows you to create new Containers already having the latest software packages installed.

Updating EZ Template Packages Inside a Container

Parallels Cloud Server allows you to update packages of the OS EZ template a Container is based on and of any application EZ templates applied to the Container. You can do it by using the `vzpkg update` utility. Assuming that Container 101 is based on the `redhat-el5-x86` OS EZ template, you can issue the following command to update all packages included in this template:

```
# vzpkg update 101 redhat-el5-x86
...
  Updating: httpd                ######################### [1/4]
  Updating: vzdev                ######################### [2/4]
  Cleanup : vzdev                ######################### [3/4]
  Cleanup : httpd                ######################### [4/4]

Updated: httpd.i386 0:2.0.54-10.2 vzdev.noarch 0:1.0-4.swsoft
Complete!
Updated:
 httpd                  i386      0:2.0.54-10.2
 vzdev                  noarch    0:1.0-4.swsoft
```

Notes:

1. Updating EZ templates is supported for running Containers only.

2. If you are going to update the cache of a commercial OS EZ template (e.g. Red Hat Enterprise Server 5 or SLES 10), you should first update software packages in the remote repository used to handle this OS EZ template and then proceed with updating the EZ template cache. Detailed information on how to manage repositories for commercial Linux distributions is provided in the *Parallels Cloud Server Templates Management Guide*.

As you can see from the example above, the `httpd` and `vzdev` applications have been updated for the `redhat-el5-x86` OS EZ template. If you wish to update all EZ templates (including the OS EZ template) inside Container 101 at once, execute this command:

```
# vzpkg update 101
...
Running Transaction
  Updating   : hwdata                 ###################### [1/2]
  Cleanup    : hwdata                 ###################### [2/2]

Updated: hwdata.noarch 0:1.0-3.swsoft
Complete!
Updated:
 hwdata                  noarch     0:0.158.1-1
```

In the example above, only the `hwdata` package inside Container 101 was out of date and updated to the latest version.

Updating OS EZ Template Caches

With the release of new updates for the corresponding Linux distribution, the created OS EZ template cache can become obsolete. Parallels Cloud Server allows you to quickly update your OS EZ template caches using the `vzpkg update cache` command.

Note: If you are going to update the cache of a commercial OS EZ template (e.g., Red Hat Enterprise Server 6 or SLES 11), you should first update software packages in the remote repository used to handle this OS EZ template and then proceed with updating the EZ template cache. Detailed information on how to manage repositories for commercial Linux distributions is provided in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

When executed, `vzpkg update cache` checks the `cache` directory in the template area (by default, the template area is located in `/vz/template`) on the Parallels server and updates all existing tarballs in this directory. However, you can explicitly indicate the tarball for what OS EZ template should be updated by specifying the OS EZ template name. For example, to update the tarball for the `centos-6-x86` OS EZ template, run this command:

```
# vzpkg update cache centos-6-x86
Loading "rpm2vzrpm" plugin
Setting up Update Process
Setting up repositories
base0             100% |=========================|  951 B    00:00
base1             100% |=========================|  951 B    00:00
```

```
base2            100% |=========================|  951 B    00:00
base3            100% |=========================|  951 B    00:00
...
```

Upon the `vzpkg update cache` execution, the old tarball is renamed by receiving the `-old` suffix (e.g., `centos-x86.tar.gz-old`):

```
# ls /vz/template/cache
centos-6-x86.tar.gz  centos-6-x86.tar.gz-old
```

You can also pass the `-f` option to `vzpkg update cache` to remove an existing tar archive and create a new one instead of it.

If the `vzpkg update cache` command does not find a tarball for one or several OS EZ templates installed on the server, it creates tar archives of the corresponding OS EZ templates and puts them to the `/vz/template/cache` directory.

CHAPTER 9

Advanced Tasks

This chapter describes those tasks that are intended for advanced system administrators who would like to obtain deeper knowledge about Parallels Cloud Server capabilities.

In This Chapter

Configuring Capabilities 168
Creating Customized Containers 170
Changing System Time from Containers 176
Obtaining Server ID from Inside a Container 177
Restarting Containers 177
Enabling VNC Access to Virtual Machines and Containers 178
Setting Immutable and Append Flags for Container Files and Directories 179
Managing iptables Modules 179
Creating Configuration Files for New Linux Distributions 181
Aligning Disks and Partitions in Virtual Machines 182
Running Parallels Cloud Server 6.0 in Virtual Machines 187
Installing Optional Parallels Cloud Server Packages 189
Working with Shared Folders in Linux Virtual Machines 189

Configuring Capabilities

Capabilities are sets of bits that permit of splitting the privileges typically held by the root user into a larger set of more specific privileges. The POSIX capabilities are defined by a draft IEEE standard (IEEE Std 1003.1e); they are not unique to Linux or Parallels Cloud Server. When the Linux or Parallels Cloud Server documentation says “requires root privileges”, in nearly all cases it really means “requires a specific capability”.

This section documents the tasks that can be achieved using per-Container capabilities in Parallels Cloud Server and all configurable capabilities.

Available Capabilities for Containers

This section lists all the capabilities that can be set with the `vzctl set` command. The capabilities are divided into two tables: the capabilities defined by the POSIX draft standard and Linux-specific capabilities. For each capability, its description is given together with the default value for a Container.

Please note that it is easy to create a non-working Container or compromise your server security by setting capabilities incorrectly. Do not change any capability for a Container without a full understanding of what this capability can lead to.

Capabilities Defined by POSIX Draft

Name	Description	Default
chown	If a process has this capability set on, it can change ownership on the files not belonging to it or belonging to another user. You have to set this capability on to allow the Container root user to change ownership on files and directories inside the Container.	on
dac_override	This capability allows to access files even if the permission is set to disable access. Normally leave this on to let the Container root access files even if the permission does not allow it.	on
dac_read_search	Overrides restrictions on reading and searching for files and directories. The explanation is almost the same as above with the sole exclusion that this capability does not override executable restrictions.	on
fowner	Overrides restrictions on setting the S_ISUID and S_ISGID bits on a file requiring that the effective user ID and effective group ID of the process shall match the file owner ID.	on
fsetid	Used to decide between falling back on the old suser() or fsuser().	on
kill	Allows sending signals to processes owned by other users.	on
setgid	Allows group ID manipulation and forged group IDs on socket credentials passing.	on
setuid	Allows user ID manipulation and forged user IDs on socket credentials passing.	on

Linux-specific Capabilities

Name	Description	Default
setpcap	Transfer any capability in your permitted set to any process ID; remove any capability in your permitted set from any process ID.	off
linux_immutable	Allows the modification of the S_IMMUTABLE and S_APPEND file attributes. These attributes are implemented only for the EXT2FS and EXT3FS Linux file systems. However, if you bind mount a directory located on the EXT2FS or EXT3FS file system into a Container and revoke this capability, the root user inside the Container will not be able to delete or truncate files with these attributes on.	on
net_bind_service	Allows to bind to sockets with numbers below 1024.	on
net_broadcast	Allows network broadcasting and multicast access.	on
net_admin	Allows the administration of IP firewalls and accounting.	off
net_raw	Allows to use the RAW and PACKET sockets.	on
ipc_lock	Allows to lock shared memory segments and mlock/mlockall calls.	on

ipc_owner	Overrides IPC ownership checks.	on
sys_module	Insert and remove kernel modules. Be very careful with setting this capability on for a Container; if a user has the permission of inserting kernel modules, this user has essentially full control over the server.	off
sys_chroot	Allows to use chroot().	on
sys_ptrace	Allows to trace any process.	on
sys_pacct	Allows to configure process accounting.	on
sys_admin	In charge of many system administrator tasks such as swapping, administering APM BIOS, and so on. Shall be set to off for Containers.	off
sys_boot	This capability currently has no effect on the Container behaviour.	on
sys_nice	Allows to raise priority and to set priority for other processes.	on
sys_resource	Override resource limits (do not confuse with user beancounters).	on
sys_time	Allows to change the system time.	off
sys_tty_config	Allows to configure TTY devices.	on
mknod	Allows the privileged aspects of mknod().	on
lease	Allows to take leases of files.	on

Creating Customized Containers

If you wish to use one or more custom applications in many identical Containers, you may want to create Containers with necessary applications already preinstalled and tuned to meet your demands.

Parallels Cloud Server offers several ways to create customized Containers with preinstalled applications:

- By creating an OS EZ template cache with preinstalled application templates.
- By making a customized base OS EZ template and using it as the basis for Containers.
- By making a non-base OS EZ template and using it as the basis for Containers.
- By making a customized application EZ template, adding it to a new configuration sample file, and using this sample file as the basis for Containers.

All these operations are described in the following subsections in detail.

Using OS Template Caches with Preinstalled Application Templates

You can use an OS EZ template cache with preinstalled application templates to quickly create multiple identical Containers without having to install applications manually or wait until they are installed automatically to each Container after its creation. The best way to create such a cache is:

1 Make a custom sample configuration file with information on the OS EZ template to cache and application EZ templates to preinstall. For example:

```
# cp /etc/vz/conf/ve-basic.conf-sample /etc/vz/conf/ve-centos-6-x86-mysql-devel.conf-sample
```

> **Note:** If you already have a custom sample configuration file created in Parallels Server Bare Metal 5 or earlier and having application EZ templates specified in it, you can reuse it instead of creating a new one.

2 Add the OS EZ template and application EZ template information to the new configuration file. Each OS and application template name must be preceded by a dot. Multiple consecutive application EZ template names must be separated by white spaces. For example:

```
# cd /etc/vz/conf
# echo 'OSTEMPLATE=".centos-6-x86"' >> ve-centos-6-x86-mysql-devel.conf-sample
# echo 'TEMPLATES=".mysql .devel"' >> ve-centos-6-x86-mysql-devel.conf-sample
```

3 Run the `vzpkg create appcache` command with your configuration file as an option. For example:

```
# vzpkg create appcache --config centos-6-x86-mysql-devel
```

> **Note:** If the resulting cache already exists, it will not be recreated and you will see a corresponding message.

The resulting archive can be found in the `/vz/template/cache` directory on the server. You can check that it exists and includes necessary application templates with the following command:

```
# vzpkg list appcache
centos-6-x86                2012-07-20 16:51:36
     mysql
     devel
```

Disabling Golden Image Functionality

The Golden Image functionality allows you to preinstall application templates to OS EZ template caches to speed up creating multiple Containers based on the same set of OS and application templates. Previously, you could either install application templates to each Container after creating it or embed them directly into a custom OS template. Golden Image is currently the easiest and fastest way to create Containers with preinstalled applications.

The Golden Image functionality is enabled by default in the `/etc/sysconfig/vz/vz.conf` global configuration file. Should you wish to disable it, do one of the following:

- Set the `GOLDEN_IMAGE` option to "`no`" in the Parallels Cloud Server global configuration file.

  The Golden Image functionality will be disabled globally.

- Set the `GOLDEN_IMAGE` option to "`no`" in the Container sample configuration file.

  The Golden Image functionality will be disabled for commands that use this specific sample configuration file.

- Create a file named `golden_image` containing "`no`" in the OS EZ template's configuration directory.

  The Golden Image functionality will be disabled for this specific OS EZ template.

- Create a file named `golden_image` containing "`no`" in the application template's configuration directory.

  The Golden Image functionality will be disabled for this specific application template, so it will not be preinstalled into any OS EZ template caches.

Using Customized OS EZ Templates

You can make a customized base OS EZ template which can then be used to create Containers with a set of application already tuned to meet your demands. To make such a template, do the following:

1. Create a metafile that will serve as the basis for your customized base OS EZ template.

   Notes:

   1. Detailed information on how to create metafiles is given in the *Parallels Cloud Server 6.0Templates Management Guide*.

   2. While creating a metafile for a new OS EZ template, make sure that the value of either the `%osname` parameter or the `%version` parameter in the metafile differs from the names or versions of all base OS EZ templates installed on the server.

2. Create one or more scripts that will be executed on different stages of the OS EZ template life cycle and customize your applications to meet your needs. For example, you can create a postinstall script with the name of `post_install.bash` and make it perform a number of customization operations on some application included in the OS EZ template after installing this application inside your Container.
3. Create a customized OS EZ template by running the `vzmktmpl` utility and passing the corresponding options to it. So, you can use the `--post-install` option and specify the path to the `post_install.bash` script from the example above to make an OS EZ template that will customize your application after installing it inside your Container.

   Note: For the full list of options allowing you to specify what scripts are to be executed on what stage of the EZ template life cycle, see **vzmktmpl** in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

4. Install the customized OS EZ template on the server using the `rpm -i` command.
5. Cache the created OS EZ template by running the `vzpkg create cache` command. Detailed information on how you can do it is provided in the *Parallels Cloud Server 6.0 Templates Management Guide*.
6. Create a Container based on the OS EZ template.

For example, to create a Container that will run CentOS 5 and have the customized `mysql` and `apache` applications installed right after its creation, you can do the following:

1. Create a metafile for the Cent OS EZ template, name it, for example, `centos_5_customized.metafile`, and save in the `/root/centos_5` directory on the server.

2 Make a script that will perform a number of custom operations after applying the `mysql` and `apache` application EZ templates to the Container, and name it `post_install.bash`.

3 Copy the script to the `/root/centos_5` directory on the server.

4 Execute the following command on the server to create the CentOS 5 OS EZ template:

```
# vzmktmpl /root/centos_5/centos_5_customized.metafile --post-install
/root/centos5/post_install.bash
```

This command creates an OS EZ template for CentOS and put it to the `/root` directory (for example, `/root/centos_customized-5-x86-ez-4.7.0-1.noarch.rpm`).

5 Install the resulting OS EZ template on the server:

```
# rpm -i /root/centos_customized-5-x86-ez-4.7.0-1.noarch.rpm
```

6 Cache the installed OS EZ template:

```
# vzpkg create cache centos_customized-5-x86
```

7 Create Container 101 on the basis of the new OS EZ template:

```
# prlctl create 101 --ostemplate centos_customized-5-x86
```

So you have just created Container 101 having the customized `mysql` and `apache` applications installed inside it.

Using EZ OS Template Sets

Another way of creating customized Containers is to make a non-base OS EZ template (also known as an OS EZ template set) differing from the corresponding base OS EZ template in the number of packages included in this template. For example, if you wish a Container to run CentOS 5 and to function as a Linux-based server only, you can create the `centos-5-x86-server` OS EZ template set and include only those packages in it that are needed for performing main server tasks. So, you can specify packages to be used for setting up file and print sharing and exclude all the packages for graphical interfaces (GNOME and KDE).

To create a non-base OS EZ template, do the following:

1 Create a metafile that will serve as the basis for your non-base OS EZ template. Any metafile for this kind of EZ template must contain the following information:

- `%osname`: the name of the Linux distribution for which you are creating the OS EZ template set. This name must correspond to that specified in the base OS EZ template. For example, if you are creating an OS template set of the base OS EZ template for CentOS 5, set the value of this parameter to `centos`.
- `%osver`: the version of the Linux distribution specified as the value of the `%osname` parameter. This name must correspond to that specified in the base OS EZ template. For example, if you are creating an OS template set of the base OS EZ template for CentOS 5, set the value of this parameter to `5`.
- `%osarch`: the system architecture where the EZ template is to be run. This name must correspond to that specified in the base OS EZ template. For example, if you are creating an OS template set of the base OS EZ template for CentOS 5, set the value of this parameter to `x86`.

- `%setname`: the name to be assigned to your non-base OS EZ template. You can specify any name you like for your OS template set:

a This name will be added to the name of the base OS EZ template after the indication of the architecture where the OS EZ template is to be run. For example, if you are creating an OS template set of the base OS EZ template for CentOS 5 that is supposed to run on x86 platforms, the name of your non-base OS EZ template should look like `centos-5-x86-`*`Template_Name`*`-ez-1.0-1.noarch.rpm`, where *`Template_Name`* is the name you specify as the value of the `%setname` parameter.

b This name will also be assigned to the directory which will store the meta data of your non-base OS EZ template after the template installation on the server. For example, it will have the name of `/vz/template/centos/5/x86/config/os/my_non_base_template` if you set the value of this parameter to `my_non_base_template`, create a non-base OS EZ template for CentOS 5, and installed it on the server.

- `%packages`: a list of RPM packages to be included in the non-base OS EZ template. This parameter allows you to specify what applications will be present inside your Containers based on this OS EZ template set right after their installation. The names of the packages listed as the value of this parameter must correspond to the names of real RPM packages (without indicating the package version, release, architecture, and the `.rpm` extension) that are stored in the repository used for managing your EZ templates.

Note: You can also specify a number of additional parameters in your metafile. For example, you may wish to add one or several extra packages to your OS EZ template set which are not available in the repository used to handle the packages for the corresponding base OS EZ template. For this purpose, you will have to specify the `%mirrorlist` parameter providing information on the repository where these extra packages are kept. Detailed information on all parameters you can set in metafiles is given in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

2 You can also (though you do not have to) create a number of scripts that will be executed on different stages of the non-base OS EZ template life cycle and customize your applications to meet your demands. The path to these scripts should then be specified after the corresponding options while creating your OS template set. For example, you can create a preinstall script with the name of `pre_install.bash` and make it perform a number of customization operations on some application included in the non-base OS EZ template before installing this application in your Container.

Note: If there are no scripts for a non-base OS EZ template, the scripts available for the corresponding base OS EZ template will be executed.

3 Create the non-base OS EZ template by running the `vzmktmpl` utility and passing the corresponding options to it, if needed. So, if you created one or several scripts in the previous step, you can use special options and specify the path to these scripts during the command execution. For example, you can use the `--pre-install` option and specify the path to the `pre_install.bash` script to make an OS EZ template that will customize your application before installing it inside your Container.

Note: For the full list of options allowing you to specify what scripts are to be executed on what stage of the EZ template life cycle, see **vzmktmpl** in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

4 Install the non-base OS EZ template on the server using the `rpm -i` command.

5 Cache the created OS EZ template by running the `vzpkg create cache` command. Detailed information on how you can do it is provided in the *Parallels Cloud Server 6.0 Templates Management Guide*.

6 Create a Container based on the OS EZ template.

Using Customized Application Templates

If the number of customized applications inside your Containers is relatively small, you can also use the following way of creating customized Containers:

1 Create a metafile that will serve as the basis for your customized application EZ template.

> **Note:** Detailed information on how to create metafiles is given in the **Creating Metafiles for EZ Templates** section of the *Parallels Cloud Server 6.0 Templates Management Guide*.

2 Create one or more scripts that will be executed on different stages of the application EZ template lifecycle and customize your applications to meet your demands. For example, you can create a postinstall script with the name of `post_install.bash` and make it perform a number of customization operations on your application after installing this application in your Container.

3 Create a customized application EZ template by running the `vzmktmpl` utility and passing the corresponding options to it. So, you can use the `--post-install` option and specify the path to the `post_install.bash` script from the example above to customize your application in accordance with your needs after installing it in your Container.

> **Note:** The full list of options allowing you to specify what scripts are to be executed on what stage of the EZ template lifecycle is provided in the **vzmktmpl** section of the *Parallels Containers 4.6 Reference Guide*.

4 Install the customized EZ template on the server using the `rpm -i` command.

5 Create a new Container configuration sample file and include the customized EZ template in this file. Detailed information on Container configuration sample files is provided in the **Managing Container Resources Configuration** section (p. 110).

6 Create a customized Container on the basis of the configuration sample.

The following example demonstrates how to create Container 101 that will run CentOS 5 and have the customized `mysql` application installed right after its creation:

1 Create a metafile for the `mysql` application, name it `mysql.metafile`, and save in the `/usr/mysql` directory on the server.

2 Make a script that will perform a number of custom operations after applying the `mysql` EZ template to the Container, and name it `post_install.bash`.

3 Copy the script to the `/usr/mysql` directory on the server.

4 Execute the following command on the server to create the `mysql` EZ template:

```
# vzmktmpl /usr/mysql/mysql.metafile --post-install /usr/mysql/post_install.bash
```

This command will create an EZ template for the `mysql` application and put it to the `/root` directory (e.g., `/root/mysql-centos-5-x86-ez-4.0.0-17.swsoft.noarch.rpm`).

5 Install the `mysql` EZ template on the server. Using the example above, you can install the template as follows:

```
# rpm -ihv /root/mysql-centos-5-x86-ez-4.0.0-17.swsoft.noarch.rpm
```

6 Create a new Container configuration sample file and add the `mysql` EZ template to a list of templates that will be installed in Containers created on the basis of this configuration sample file.

7 Create Container 101 by using the `prlctl create` command and the `mysql` sample file:

```
# prlctl create 101 --ostemplate centos-5-x86
```

So, you have just created Container 101 that already has the customized `mysql` application installed.

Changing System Time from Containers

Normally, it is impossible to change the system time from a Container. Otherwise, different Containers could interfere with each other and could even break applications depending on the system time accuracy.

Normally only the server system administrator can change the system time. However, if you want to synchronize the time via Network Time Protocol (NTP), you have to run NTP software, which will connect to external NTP servers and update the system time. It is not advisable to run application software on the server itself, since flaws in the software can lead to compromising all Containers on this server. Thus, if you plan to use NTP, you should create a special Container for it and configure it to have the `sys_time` capability. The example below illustrates configuring such a Container:

```
# prlctl set 101 --capability sys_time:on
Unable to set capability on running Container
Saved parameters for Container 101
```

The output of the above command warns you that `prlctl` cannot apply changes in the capabilities to a running Container. The Container has to be restarted before changes take effect:

Note: Currently, the `--capability` option may not be supported by the `prlctl` utility. Use `vzctl` instead.

```
# prlctl stop 101; prlctl start 101
# ssh root@ct101
root@ct101's password:
Last login: Mon Feb 28 23:25:58 2007 from 10.100.40.18
[root@ct101 root]# date
Mon Feb 28 23:31:57 EST 2007
[root@ct101 root]# date 10291300
Tue Feb 29 13:00:00 EST 2007
[root@ct101 root]# date
Tue Feb 29 13:00:02 EST 2007
[root@ct101 root]# logout
Connection to Container101 closed.
```

```
# date
Tue Feb 29 13:01:31 EST 2010
```

The command session above shows the way to change the system time from Container 101. The changes will affect all the Containers and the server itself. It is not advisable to have more than one Container with the `sys_time` capability set on.

NTP is described in Internet Standard RFC 1305; more information including client software can be obtained from the NTP web server (http://www.ntp.org).

Obtaining Server ID from Inside a Container

The default Parallels Cloud Server installation does not allow users inside a Container to obtain any information specific to the Parallels server the Container is running on. The reason is that no Container shall have knowledge about the corresponding server. A Container can be transparently migrated to another server, and if this Container runs any applications depending on the particular server, these applications might fail after the migration.

In some situations, however, you need to provide a unique server ID to some applications. For example, you might want to license your application per server. In this case, after the migration your customer will need to re-apply the license for your application.

Parallels Cloud Server provides access to the unique server ID via the `/proc/vz/hwid` file. The default Parallels Cloud Server installation makes this file accessible to Containers from 1 to 100 (i.e. Containers with reserved IDs). It is possible to change this range in the global configuration file (`vz.conf`). For example, this is the way to make the file visible in Containers from 1 to 1000:

```
# vi /etc/vz/vz.conf
VZPRIVRANGE="1 1000"
# prlctl exec 101 cat /proc/vz/hwid
0C3A.14CB.391B.6B69.02C9.4022.3E2F.CAF6
```

The above example illustrates accessing the server ID from Container 101.

Restarting Containers

To restart Containers in Parallels Cloud Server, Container administrators can use the same commands they normally use on normal Linux systems. For example, administrators can run the `reboot` or `shutdown -r` commands from inside Containers to restart them.

If necessary, you can forbid administrators to initiate reboots from inside Containers:

- To disable reboots for a specific Container, add the `ALLOWREBOOT="no"` line to the Container configuration file (`/etc/vz/conf/<CT_ID>.conf`).
- To disable reboots globally for all Containers on the server, add the `ALLOWREBOOT="no"` line to the global configuration file (`/etc/vz/vz.conf`).

- To disable reboots globally but enable them for specific Containers, add the `ALLOWREBOOT="no"` line to the global configuration file (`/etc/vz/vz.conf`) and explicitly specify `ALLOWREBOOT="yes"` in the configuration files of the respective Containers.

Enabling VNC Access to Virtual Machines and Containers

You can use your favorite VNC clients to connect to and manage Containers and virtual machines. To do this, you need to complete these steps:

1. Enable VNC access in the desired virtual machine (p. 178) or Container (p. 178).
2. Connect to the virtual machine or Container with a VNC client (p. 179).

The sections below describe both steps in details.

Enabling VNC Access to Virtual Machines

To enable VNC access to a virtual machine, you need to do the following:

1. Enable VNC support in the virtual machine.
2. Specify the TCP port number on the physical server that will be used to listen to VNC connections for the virtual machine.

   Note: A unique port number must be specified for each virtual machine where you plan to connect via VNC.

3. Set a password to secure your VNC connection.

You can perform all these operations with a single command. For example:

```
# prlctl set MyVM --vnc-mode manual --vnc-port 5901 --vnc-passwd XXXXXXXX
```

The changes will come into effect on the next virtual machine start.

Enabling VNC Access to Containers

To enable VNC access to a Container, you need to do the following:

1. Make sure you have a valid user account in the Container to be able to log into it.
2. Make sure the Container is running.
3. Set the VNC mode and password for the Container. For example:

```
# prlctl set 101 --vnc-mode manual --vnc-port 6501 --vnc-passwd XXXXXXXX
```

   Note: A port number must be unique for each Container you open VNC access to. In the auto mode, correct port numbers are assigned automatically. In the manual mode, you need to make sure port numbers are unique yourself.

Connecting with a VNC Client

After you have enabled VNC access to the virtual machine/Container, you can connect to it with your favorite VNC client. To do this, you need to pass the following parameters to the VNC client:

- IP address of the server where the virtual machine/Container is hosted.
- Port number and password you specified when enabling VNC access.
- Valid user account in the virtual machine/Container.

Setting Immutable and Append Flags for Container Files and Directories

You can use standard Linux utilities—`chattr` and `lsattr`–to set extra flags for files and directories inside your Containers and to query their status, respectively. Currently, two of these extra flags—'append' and 'immutable'—are supported. For example, you can execute the following command to set the 'immutable' flag for the `/root/MyFile` file inside Container 101:

```
[root@ct101 root] chattr +i /root/MyFile
```

To check that the 'immutable' flag has been successfully set, use the following command:

```
[root@ct101 root] lsattr /root/MyFile
----i--------- /root/MyFile
```

Note: For detailed information on the `chattr` and `lsattr` utilities, see their manual pages.

Managing iptables Modules

This section describes the process of managing iptables modules for both physical servers and Containers.

Using iptables Modules on Servers

The process of filtering network packets on systems running Parallels Cloud Server does not differ from that on standalone Linux servers. You can use the standard *iptables* tool to control how network packets enter, move through, and exit the network stack within the Parallels Cloud Server kernel.

For your reference, below are several resources you can consult to get detailed information on using `iptables` on Linux servers:

- *Red Hat Enterprise Linux 6 Security Guide* (http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Security_Guide/index.html) contains a section focusing

on packet filtering basics and explaining various options available for `iptables`.

- *Iptables Tutorial 1.2.2* (http://www.frozentux.net/iptables-tutorial/iptables-tutorial.html) explains in great detail how `iptables` is structured and how it works.

Using iptables Modules in Containers

Using `iptables` modules in Containers requires additional configuration on your part.

Defining the basic set of iptables modules for Containers

All modules you plan to use in a Container must be loaded on the physical server where the Container is hosted. So if you try to create in the Container a rule for a module that is not loaded on the server, `iptables` will return an error.

You define which `iptables` modules are allowed for use in Containers by using the `iptables` service of the physical server. To tell this service to load on the server the set of modules you plan to use in Containers, edit the `/etc/sysconfig/iptables-config` file and specify the desired modules as the value of the `IPTABLES_MODULES` parameter in this file.

Let us assume that you want to make the following modules automatically load on the server startup, thus allowing their use in Containers: `ipt_REJECT`, `iptable_filter`, `iptable_mangle`, `xt_length`, `xt_hl`, `xt_tcpmss`, `xt_TCPMSS`, `xt_multiport`, `xt_limit`, `xt_dscp`. To do this:

1. Open the `/etc/sysconfig/iptables-config` file for editing.
2. Set the value of the `IPTABLES_MODULES` parameter to the following:

```
IPTABLES_MODULES="ipt_REJECT iptable_filter iptable_mangle xt_length xt_hl xt_tcpmss
xt_TCPMSS xt_multiport xt_limit xt_dscp"
```

3. Restart the `iptables` service for the changes to the `/etc/sysconfig/iptables-config` file to come into effect:

```
# service iptables restart
```

> **Note:** After loading some modules on the server, you may need to perform a number of additional configuration steps and restart a Container to allow the use of these modules in the Container (for details, see the information below).

Using Parallels Cloud Server settings to configure iptables modules

You can use two configuration settings specific to Parallels Cloud Server to additionally define the set of `iptables` modules allowed for use in Containers:

- `IPTABLES` in the global configuration file (`/etc/vz/vz.conf`). This parameter, if set, is valid for all Containers on the server.
- `IPTABLES` in the configuration file of specific Containers (`/etc/vz/conf/<CT_ID>.conf`). This parameter is empty by default so that the settings from the `/etc/vz/vz.conf` global

configuration file apply to all Containers. Setting the IPTABLES parameter in the configuration file of a specific Container will override the IPTABLES parameter in /etc/vz/vz.conf

Using IPTABLES parameters, you can forbid the use of some iptables modules in Containers. The list of such parameters is given below:

- iptable_filter: If set, this parameter allows the use of rules in both IPv4 and IPv6 filter tables in Containers.
- iptable_mangle: If set, this parameter allows the use of rules in both IPv4 and IPv6 mangle tables in Containers.
- ip_conntrack: If set, this parameter allows the use of conntrack rules in any tables in Containers.
- iptable_nat: If set, this parameter allows the use of rules in the NAT table in Containers. If iptable_nat is set, it also enables all conntrack-related modules even if ip_conntrack is not turned on.

Notes:

1. By default, the NAT table and conntrack rules are disabled in Containers. For information on how to enable them, see **Using conntrack Rules and NAT Tables in Containers** below.

2. If the IPTABLES parameters in the global and Container configuration files are not defined, the in-kernel default allows the use of iptable_filter and iptable_mangle modules in Containers, but forbids the use of ip_conntrack and iptable_nat modules.

All modules listed in the IPTABLES parameter in the /etc/vz/vz.conf global configuration file are loaded on the server during the vz service start, even if they are not listed in the /etc/sysconfig/iptables-config file. The default setting of IPTABLES in /etc/vz/vz.conf is configured to load typical iptables rules to allow their use in Containers, without making any changes to the iptables service of the physical server.

Using conntrack Rules and NAT Tables in Containers

By default, the NAT table and conntrack rules are disabled and not allowed for use in Containers even if they are loaded on the server. To allow their use in Containers, do one of the following:

- Enable this functionality for all Containers on a server. To do this, open the /etc/vz/vz.conf global configuration file for editing and specify ip_conntrack and iptable_nat as the value of the IPTABLES parameter.
- Enable this functionality for a specific Container. To do this, open the configuration file of the Container (/etc/vz/conf/*<CT_ID>*.conf) for editing and specify ip_conntrack and iptable_nat as the value of the IPTABLES parameter.

Creating Configuration Files for New Linux

Distributions

Distribution configuration files are used to distinguish among Containers running different Linux versions and to determine what scripts should be executed when performing the relevant Container-related operations (e.g., assigning a new IP address to the Container). For details on distribution configurations files, see **Linux Distribution Configuration Files** in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

All Linux distributions shipped with Parallels Cloud Server have their own configuration files located in the `/etc/vz/conf/dists` directory on the Parallels server. However, you may wish to create your own distribution configuration files to support new Linux versions released. Let us assume that you wish your Containers to run the CentOS 5 Linux distribution and, therefore, have to make the `centos-5.conf` distribution configuration file to define what scripts are to be executed while performing major tasks with Containers running this Linux version. To do this:

1. In the Container configuration file (with the name of `/etc/vz/conf/<CT_ID>.conf`), specify `centos-5` as the value of the `DISTRIBUTION` variable (for example, `DISTRIBUTION="centos-5"`).
2. Create the `centos-5.conf` configuration file in the `/etc/vz/conf/dists` directory. The easiest way to do it is copy one of the existing configuration files by executing the following command in the `/etc/vz/conf/dists` directory:

```
# cp fedora.conf centos-5.config
```

   In the example above, we assume that the `fedora.conf` file is present in the `/etc/vz/conf/dists` directory on the Parallels server. In case it is not, you may use any other distribution configuration file available on your server.

3. Open the `centos.conf` file for editing with the help of any text editor:

```
# vi centos-5.conf
```

4. In the `centos-5.conf` file, go to the first entry and, in the right part of the entry, specify the name of the script you wish to be run on issuing the `prlctl` command with the parameter specified in the left part of the entry. For example, if you wish the script to be executed while assigning a new IP address to your Container and the script has the `my_centos_script` name, your entry should look as follows:

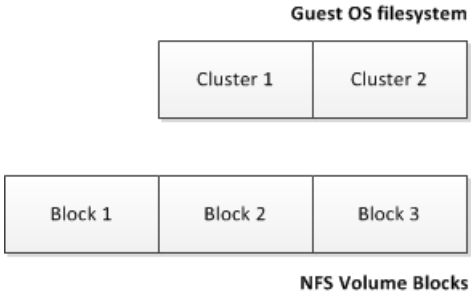
```
ADD_IP=my_centos_script-add_ip.sh
```

   Note: All acceptable parameters are described in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

5. Repeat **Step 4** for all entries in the file.
6. Place the scripts for the new Linux distribution to the `/etc/vz/conf/dists/scripts` directory on the Node. Make sure the names of these scripts coincide with those specified in the `centos-5.conf` file.

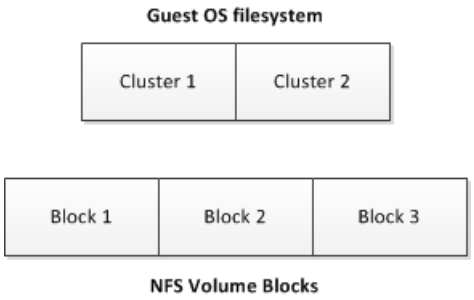
Aligning Disks and Partitions in Virtual Machines

Most of the modern operating systems (Windows Server 2008 or Red Hat Enterprise Linux 6) automatically align the partitions when you install them in virtual machines. For example, Windows Server 2008 creates a default partition offset of 1024 KB to satisfy the partition alignment requirements. The following figure shows an example of the correct partition alignment:



In this example, any cluster (the smallest unit of data) in the guest OS file system is aligned with the boundaries of an NFS block, and reading from or writing to a cluster requires only access to one NFS block. For example, reading from Cluster 1 causes only a read from Block 1.

At the same time, virtual machines running non-modern systems (for example, Windows Server 2003 or Red Hat Enterprise Linux 5) do usually have misaligned partitions, which is shown in the figure below:



In this example, clusters of the guest OS file system do not match the boundaries of NFS blocks, and reading from or writing to a cluster requires access to several NFS blocks. For example, reading from Cluster 1 causes two reads: from Block 1 and from Block 2. This results in a slower read time as compared to properly aligned partitions and leads to performance degradation.

Aligning partitions

Basically, to align disks and partitions in virtual machines, you need to set an offset so that clusters in the guest OS file system match the volume block size on your NFS storage. Usually, the block size of most network storages is 512 bytes or a multiple of 512 bytes. As an example, the following sections describe the procedure of aligning disks and partitions for Linux and Windows virtual machines assuming that the size of your NFS blocks is 512 bytes.

When deciding on aligning disks and partitions, take into account that this process destroys all data on these disks and partitions. So if you want to have a correctly aligned system partition, you need

to align your disks and partitions before creating a virtual machine and installing a guest operating system in it. If you do not want an aligned system partition, you can first create a virtual machine and install a guest OS in it, and then align your data disks from inside the virtual machine.

The sections below demonstrate how to align disks and partitions before you start installing a guest OS. You can, however, use a similar procedure to align data disks and partitions from inside your virtual machines.

Checking partition alignment in existing virtual machines

First of all, you may wish to know how you can check that the partitions of a virtual machine are not aligned. Depending on the operating system installed in the virtual machine, you can do the following.

Linux virtual machines

To check the partition alignment in a Linux virtual machine, log in to this virtual machine and run the following command:

```
# fdisk -l -u /dev/device_name
```

For example, to check the partition alignment on the `sdc` device, you can run this command:

```
# fdisk -l -u /dev/sdc

Disk /dev/sdc: 73.0 GB, 73014444032 bytes
255 heads, 63 sectors/track, 8876 cylinders, total 142606336 sectors
Units = sectors of 1 * 512 = 512 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1   *          63      208844      104391   83  Linux
/dev/sdc2          208845   142592939    71192047+  8e  Linux LVM
```

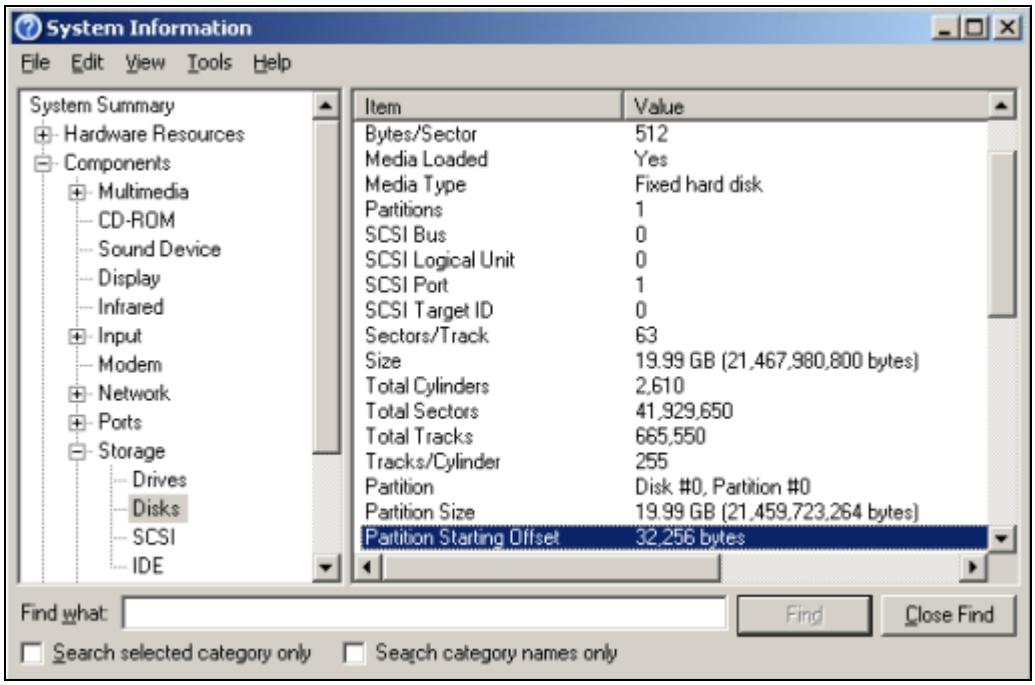
Pay attention to the number of sectors in the **Start** column. Usually, a sector contains 512 bytes, which makes up 32256 bytes for 63 sectors for the `/dev/sdc1` partition and 26105625 bytes for 208845 for the `/dev/sdc2` partition. For a partition to be properly aligned, it must align with 4096 byte boundaries (assuming that the block size of your storage is 4 KB). As 32256 and 106928640 is not a multiple of 4096, the partitions `/dev/sdc1` and `/dev/sdc2` are not aligned properly. To align them, you should offset

- the `/dev/sdc1` partition by 1 sector so that it starts at 64. In this case, 64 sectors each containing 512 bytes make up 32768 that is a multiple of 4096.
- the `/dev/sdc2` partition by 3 sectors so that it starts at 208848. In this case, 208848 sectors each containing 512 bytes make up 106930176 that is a multiple of 4096.

Windows virtual machines

To check the partition alignment in a Windows virtual machine, do the following:

1. Click **Start** > **Run**, type `msinfo32.exe`, and press Enter to open System Information.
2. Navigate to **Components** > **Storage** > **Disks**, and look for the **Partition Starting Offset** field in the right part of the window.



To find out if the partition is aligned properly, use the method described above for Linux virtual machines.

Aligning disks for Linux virtual machines

To align partitions for use in a Linux virtual machine, you need a working Linux virtual machine. Once you have it at hand, follow the steps below:

1. Create a new disk for the virtual machine.

   On this disk, you will create aligned partitions. Then you will connect the disk to a new virtual machine and install your Linux guest OS on this disk.
2. Start the virtual machine and log in to it using SSH.
3. Run the `fdisk` utility for the disk you want to align.
4. Create a primary partition, and set the starting block number for the created partition.
5. Repeat steps 4-6 to create and align all partitions you plan to have in your new virtual machine.

The following example creates partition #1 with the size of 1 GB on the `/dev/sda` device and uses the offset of 64 KB.

```
# fdisk /dev/sda

Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.
```

```
The number of cylinders for this disk is set to 1044.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First sector (63-16777215, default 63): 64
Last sector or +size or +sizeM or +sizeK (64-16777215, default 16777215): 208848

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

Once you align all the necessary partitions, disconnect the disk from the virtual machine. When creating a new virtual machine, choose this disk for use with this virtual machine.

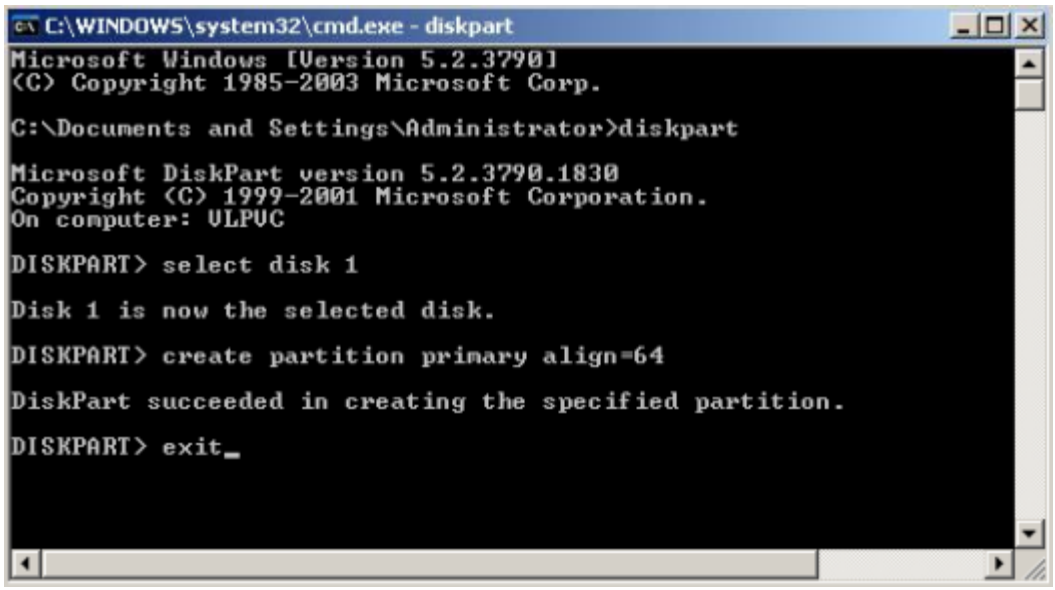
Aligning partitions for Windows virtual machines

To align a disk for a Windows virtual machine, you need a working Windows virtual machine. Once you have it at hand, you can use the `diskpart` or `diskpar` utility (depending on your operating system) to align the disk:

1. Create a new disk for the virtual machine.

   On this disk, you will create aligned partitions. Then you will connect the disk to a new virtual machine and install your Windows guest OS on this disk.
2. Open the command-line prompt, and run the `diskpart` or `diskpar` utility.
3. Select the disk you want to align.
4. Create the primary partition on the disk, and align it.
5. Exit the `diskpart` utility, and close the command-line prompt.

The following example demonstrates how to use the `diskpart` utility to align disk 1 by setting the offset of 64 for it:


C:\WINDOWS\system32\cmd.exe - diskpart
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.
C:\Documents and Settings\Administrator>diskpart
Microsoft DiskPart version 5.2.3790.1830
Copyright (C) 1999-2001 Microsoft Corporation.
On computer: VLPVC
DISKPART> select disk 1
Disk 1 is now the selected disk.
DISKPART> create partition primary align=64
DiskPart succeeded in creating the specified partition.
DISKPART> exit

Once you align the virtual disk, disconnect it from the virtual machine. When creating a new virtual machine, choose this disk for use with this virtual machine.

Creating a template of a virtual machine with aligned partitions

To facilitate the procedure of creating virtual machines that have aligned system partitions, you can create a template of the aligned virtual machine and deploy new virtual machines from this template.

For example, if you align a disk by following the steps in **Aligning partitions for Windows virtual machines**, then create a new virtual machine that uses this disk, and then install Windows Server 2003 operating system in the virtual machine, you will have a clean Windows Server 2003 installation on the correctly aligned disk. Now you can create a template of this virtual machine and use this template each time you need to deploy a new virtual machine with Windows Server 2003.

Running Parallels Cloud Server 6.0 in Virtual Machines

Starting with version 5, you can install Parallels Cloud Server in virtual machines. Running Parallels Cloud Server 6.0 in a virtual machine may prove useful if you want to evaluate the product but do not have a spare physical server.

To run virtual machines with Parallels Cloud Server 6.0, a physical server must meet the following requirements:

Have the following architecture:

- Intel with VT-x and EPT (Nehalem, Westmere, SandyBridge)

- AMD SVM and RVI (Barcelona, MagnyCore, Bulldozer)

Run one of the following products:

- Parallels Desktop for Mac 7
- Parallels Workstation for Windows 6
- Parallels Workstation for Linux 6
- VMware Fusion 3

Recommended Virtual Machine Configuration

A virtual machine is best optimized for use with Parallels Cloud Server 6.0 if it has the following configuration:

- CPU:

  2 or more virtual CPUs

- Memory:

  2 GB or more RAM

- Hard disk:

  40 GB or more on a plain hard disk

Installing Parallels Cloud Server in a virtual machine does not differ from installing it on a standalone server. For detailed installation instructions, consult the *Parallels Cloud Server 6.0 Installation Guide*.

Restrictions and Peculiarities

When using Parallels Cloud Server in a virtualized environment, keep in mind the following restrictions and specifics:

- Running Parallels Cloud Server in a virtual machine is intended for evaluation purposes only. You are not recommended to use such installations in production.
- If you change the configuration of a virtual machine where Parallels Cloud Server is installed, you may need to reactivate the product.
- When you start a virtual machine with Parallels Cloud Server, VMware Fusion may warn you that it requires full access to the network traffic. Ignore this message, and proceed with booting the virtual machine.
- To run in a virtualized Parallels Cloud Server environment, a virtual machine must have Parallels Tools installed. Before starting a virtual machine for the first time, make sure that you have installed Parallels Tools in it.
- Virtual machines may run only 32-bit operating systems and have only one CPU.

Installing Optional Parallels Cloud Server Packages

Parallels Cloud Server comes with everything you may need already installed. However, you can also install optional Parallels Cloud Server packages from the Parallels remote repositories by means of the `yum` command. For example:

```
# yum install pctl
...
Installed:
  pctl.x86_64 1:4.0.0.18

Complete!
```

Note: For more information on using `yum` in Parallels Cloud Server, see **Updating Parallels Cloud Server** (p. 162) and the `yum` Linux manual page.

Working with Shared Folders in Linux Virtual Machines

The Shared Folders functionality allows you to share files and folders between the physical server and virtual machines. When you share a folder with a virtual machine, this folder and all its files become visible in the virtual machine and can be accessed by users of this virtual machine. The ownership and permissions of the shared folder and files depend on the sharing options set in your system. In general, you have the following options for sharing folders and files:

- Sharing folders with ownership of a virtual machine user (p. 189).
- Mounting shared folders as a specific user (p. 191).
- Sharing folders with ownership of a user on the physical server (p. 191).

All options are described in the following sections in detail.

Notes:

1. Shared folders are not a POSIX-compliant file system and are designed just for sharing files and folders.

2. For the Shared Folders functionality to work, Parallels Tools must be installed in a virtual machine.

Sharing with Ownership of a Virtual Machine User

By default, Shared Folders are set to work in such a way that any user currently logged in to a virtual machine is considered as the owner of a shared folder and all files in it. To check that this

mode of operation is enabled for a virtual machine, log in to it and examine the `/etc/fstab` file. If this mode is on, the following lines should be present in `/etc/fstab`:

```
# Parallels Shared Folder mount
none  /media/psf   prl_fs
sync,nosuid,nodev,noatime,share,context=system_u:object_r:removable_t:s0,nofail
```

In general, to share a folder on the physical server with a virtual machine, you need to do the following:

1. Make sure that the virtual machine has Parallels Tools installed.
2. Enable the Shared Folders functionality for the virtual machine.
3. Specify what folder to share and under which name to mount it in the virtual machine.

Once you complete these steps, the shared folder should appear as a subdirectory in `/media/psf` in the virtual machine and can be accessed by users of this virtual machine.

For example, to share the `/home/SharedFolder` directory located on the server and storing two files (`1.txt` and `2.txt`) and mount it to the `/media/psf/Shared` directory in the `MyVM` virtual machine, you can run these commands:

```
# prlctl set MyVM --shf-host on
# prlctl set MyVM --shf-host-add Shared --path /home/SharedFolder
```

After that, the `Shared` folder should appear in the `/media/psf` directory in the `MyVM` virtual machine and contain the same two files as the `/home/SharedFolder` directory on the server:

```
# ls -l /home/SharedFolder
total 4
-rw-r--r-- 1 root root 143 May 4 19:04 1.txt
-rw-r--r-- 1 root root   0 May 4 19:04 2.txt

[bob@dhcp-10-30-21-207] ls -l /media/psf/Shared
total 4
-rw-r--r--. 1 bob bob 143 May 4 19:04 1.txt
-rw-r--r--. 1 bob bob   0 May 4 19:04 2.txt
```

As you can see, the files are exactly the same but owned by different users. While the owner of the files on the server is `root`, the owner of the same files in the `MyVM` virtual machine is `bob`, the user currently logged in to this virtual machine.

If you now log off of the `MyVM` virtual machine and log in to it again as the `jack` user, the owner of `1.txt` and `2.txt` will be changed to `jack`:

```
[jack@dhcp-10-30-21-207]$ ls -l /media/psf/Shared
total 4
-rw-r--r--. 1 jack jack 143 May 4 19:04 1.txt
-rw-r--r--. 1 jack jack   0 May 4 19:04 2.txt
```

As for permissions for the `/media/psf/Shared` directory and its files, they may or may not coincide with those of these files on the physical server.

To unmount the `Shared` folder from the `MyVM` virtual machine, use this command:

```
# prlctl set MyVM --shf-host-del Shared
```

Mounting as a Specific User

You can also configure the Shared Folders functionality in such a way that the owner of a shared folder and all its files will be the user who mounts the folder. To enable this option for your system:

1. Open the `/etc/fstab` file for editing.
2. Locate the Shared Folders line:

```
none  /media/psf   prl_fs
sync,nosuid,nodev,noatime,share,context=system_u:object_r:removable_t:s0,nofail
```

3. Remove the `share` option from this line so it looks like below:

```
none  /media/psf   prl_fs
sync,nosuid,nodev,noatime,context=system_u:object_r:removable_t:s0,nofail
```

4. Save the file.

Once you complete the steps above, shared folders in a virtual machine will be seen by default as mounted and, therefore, owned by the `root` user, regardless of the user currently logged in to the virtual machine. For example:

```
[bob@dhcp-10-30-21-207]$ ls -l /media/psf/Shared
total 4
-rw-r--r--. 1 root root 143 May 4 19:04 1.txt
-rw-r--r--. 1 root root   0 May 4 19:04 2.txt
```

If necessary, you can change the owner and group of the `/media/psf/Shared` folder by mounting it with specific UID and GID options.

Sharing with Ownership of a Server User

Another way of using the Shared Folders functionality is to configure your system so that a shared folder has the same owner and group as it has on the physical server. To enable this option for your system:

1. Open the `/etc/fstab` file for editing.
2. Locate the Shared Folders line:

```
none  /media/psf   prl_fs
sync,nosuid,nodev,noatime,share,context=system_u:object_r:removable_t:s0,nofail
```

3. Replace the `share` option with the `plain` one, or if the former is not present, just add the `plain` option to the line:

```
none  /media/psf   prl_fs
sync,nosuid,nodev,noatime,plain,context=system_u:object_r:removable_t:s0,nofail
```

4. Save the file.

Now if you share the `/home/SharedFolder` directory that contains the files `1.txt` and `2.txt` and is owned by the `greg` user and the `greg` group and mount it to the `/media/psf/Shared` directory in the `MyVM` virtual machine, both files will have `greg` as the owner and group owner in the virtual machine:

```
# ls -l /home/SharedFolder
```

```
total 4
-rw-r--r-- 1 greg greg 143 May 4 15:15 1.txt
-rw-r--r-- 1 greg greg   0 May 4 19:04 2.txt

[bob@dhcp-10-30-21-207]$ ls -l /media/psf/Shared
total 4
-rw-r--r--. 1 greg greg 143 May 4 19:04 1.txt
-rw-r--r--. 1 greg greg   0 May 4 19:04 2.txt
```

CHAPTER 10

Legacy Features

This chapter provides brief information on VZFS, a legacy file system used in earlier versions of Parallels Server Bare Metal, and explains the way of configuring your system for work with legacy Containers.

In This Chapter

Using Virtuozzo File System 193
Creating VZFS-based Containers 193
Converting VZFS Containers to New Layout 194
Listing Legacy Container Backups 194

Using Virtuozzo File System

Virtuozzo File System (VZFS) is a legacy file system that allows sharing common files among multiple Containers without sacrificing flexibility. Container users can modify, update, replace, and delete shared files. When a user modifies a shared file, VZFS creates a private copy of that file transparently for the user. Thus, modifications do not affect other users of the same file.

Although VZFS can help you save disk space and memory, it also has a number of limitations:

- You cannot store Containers using VZFS in Parallels Cloud Storage clusters.
- To migrate or restore a Container, you always need to have a corresponding OS template installed on the destination server.
- VZFS-based Containers lack some functionality provided by Parallels Cloud Server 6.0 (like creating and managing snapshots).

Note: For more information on VZFS, see the documentation for Parallels Server Bare Metal 5.0.

Creating VZFS-based Containers

When you install Parallels Cloud Server 6.0, your system is automatically configured to use the new Container-in-a-file layout for Containers. So when you create a new Container, all its files are put to a single image, similar to a virtual machine's hard disk.

If you want to create a legacy Container using the VZFS file system, you first need to complete these tasks:

1. Set the `VEFSTYPE` parameter to `vz4` in the global configuration file `/etc/vz/vz.conf`.
2. Create a VZFS-compatible OS template cache using the `vzpkg create cache` command. For example, to create an updated cache for the CentOS 6 template, run this command:

```
# vzpkg create cache centos-6-x86
```

3. Once the cache is updated, you can create a Container in the usual way using the `prlctl create` command, for example:

```
# prlctl create 101 --ostemplate centos-6-x86
```

Note: For more information on managing legacy Containers, see the documentation for Parallels Server Bare Metal 5.0.

Converting VZFS Containers to New Layout

To convert a Container based on VZFS to the Container-in-an-image-file layout used in Parallels Cloud Server 6.0:

1. Make sure that the `/vz` partition is formatted as ext4.

   Using Containers with the Container-in-an-image-file layout is supported only for `/vz` partitions formatted as ext4.

2. Use the `vzctl convert` command to convert the Container. For example, to convert Container 101, you can run this command:

```
# vzctl convert 101 --velayout 5
```

   If the `--velayout` option is omitted, the command converts a Container to the layout defined by the `VEFSTYPE` parameter in the global configuration file `/etc/vz/vz.conf`. For the Container-in-an-image-file layout, this parameter should be set to `ext4`.

Listing Legacy Container Backups

After upgrading to Parallels Cloud Server 6.0, the `pbackup list` command may not list old backups made in Parallels Virtuozzo Containers or the previous version of Parallels Cloud Server.

To enable listing of old backups, uncomment the `BACKUP_COMPATIBILITY_MODE` parameter in the `/etc/vzbackup.conf` configuration file and set its value to `yes`.

CHAPTER 11

Troubleshooting

This chapter provides the information about those problems that may occur during your work with Parallels Cloud Server and suggests the ways to solve them, including getting technical support from Parallels.

In This Chapter

General Considerations 195
Kernel Troubleshooting..... 197
Problems with Container Management..... 199
Getting Technical Support 201

General Considerations

The general issues to take into consideration when troubleshooting your system are listed below. You should read them carefully before trying to solve more specific problems.

- Make sure a valid license is always loaded on the server. If your license has expired and the grace period is over, all the virtual machines and Containers on your server will be stopped.
- You should always remember where you are currently located in your terminal. Check it periodically using the `pwd`, `hostname`, `ifconfig`, `cat /proc/vz/veinfo` commands. One and the same command executed inside a virtual machine and Container and on the server can lead to very different results. You can also set up the `PS1` environment variable to show the full path in the `bash` prompt. To do this, add these lines to `/root/.bash_profile`:

```
PS1="[\u@\h \w]$ "
export PS1
```

- If the server slows down, use `vmstat`, `ps` (`ps axfw`), `dmesg`, `top` (`vztop`) to find out what is happening, never reboot the machine without investigation. If no thinking helps restore the normal operation, use the Alt+SysRq sequences to dump the memory (`showMem`) and processes (`showPc`).
- If the server was incorrectly brought down, on its next startup all the partitions will be checked and quota recalculated for each Container, which dramatically increases the startup time.
- Do not run any binary or script that belongs to a Container directly from the server, for example, do not ever do that:

```
cd /vz/root/99/etc/init.d
./httpd status
```

Any script inside a Container could have been changed to whatever the Container owner chooses: it could have been trojaned, replaced to something like `rm -rf`, etc. You can use only `prlctl exec/prlctl enter` to execute programs inside a Container.

- Do not use init scripts on the server. An init script may use `killall` to stop a service, which means that all similar processes will be killed in all Containers. You can check `/var/run/Service.pid` and kill the correspondent process explicitly.
- You must be able to detect any rootkit inside a Container. It is recommended to use the `chkrootkit` package for detection (you can download the latest version from www.chkrootkit.org), or at least run

```
rpm -Va|grep "S.5"
```

to check up if the MD5 sum has changed for any RPM file.

You can also run `nmap`, for example:

```
# nmap -p 1-65535 192.168.0.1

Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
Interesting ports on  (192.168.0.1):
(The 65531 ports scanned but not shown below are in
  state: closed)
Port       State       Service
21/tcp     open        ftp
22/tcp     open        ssh
80/tcp     open        http
111/tcp    open        sunrpc

Nmap run completed -- 1 IP address (1 host up) scanned
  in 169 seconds
```

to check if any ports are open that should normally be closed.

That could however be a problem to remove a rootkit from a Container and make sure it is 100% removed. If you're not sure, create a new Container for that customer and migrate his/her sites and mail there.

- Check the `/var/log/` directory on the server to find out what is happening on the system. There are a number of log files that are maintained by the system and Parallels Cloud Server (the `boot.log`, `messages`, etc.), but other services and programs may also put their own log files here depending on your distribution of Linux and the services and applications that you are running. For example, there may be logs associated with running a mail server (the `maillog` file), automatic tasks (the `cron` file), and others. However, the first place to look into when you are troubleshooting is the `/var/log/messages` log file. It contains the boot messages when the system came up as well as other status messages as the system runs. Errors with I/O, networking, and other general system errors are reported in this file. So, we recommend that you read to the `messages` log file first and then proceed with the other files from the `/var/log/` directory.
- Subscribe to bug tracking lists. You should keep track of new public DoS tools or remote exploits for the software and install them into Containers or at servers.
- When using `iptables`, there is a simple rule for Chains usage to help protect both the server

and its Containers:

- use INPUT, OUTPUT to filter packets that come in/out the server
- use FORWARD to filter packets that are designated for Containers

Kernel Troubleshooting

Using ALT+SYSRQ Keyboard Sequences

Press ALT+SYSRQ+H (3 keys simultaneously) and check what is printed at the server console, for example:

```
SysRq: unRaw Boot Sync Unmount showPc showTasks showMem loglevel0-8 tErm kIll killalL
Calls Oops
```

This output shows you what ALT+SYSRQ sequences you may use for performing this or that command. The capital letters in the command names identify the sequence. Thus, if there are any troubles with the machine and you're about to reboot it, please press the following sequences before pressing the **Power** button:

ALT+SYSRQ+M to dump memory info

ALT+SYSRQ+P to dump processes states

ALT+SYSRQ+S to sync disks

ALT+SYSRQ+U to unmount filesystems

ALT+SYSRQ+L to kill all processes

ALT+SYSRQ+U try to unmount once again

ALT+SYSRQ+B to reboot

If the server is not rebooted after that, you can press the **Power** button.

Saving Kernel Faults (OOPS)

You can use the following command to check for the kernel messages that should be reported to Parallels Cloud Server developers:

```
grep -E "Call Trace|Code" /var/log/messages*
```

Then, you should find kernel-related lines in the corresponding log file and figure out what kernel was booted when the oops occurred. Search backward for the "Linux" string, look for strings like that:

```
Sep 26 11:41:12 kernel: Linux version 2.6.18-8.1.1.el5.028stab043.1 (root@rhel5-32-
build) (gcc version 4.1.1 20061011 (Red Hat 4.1.1-30)) #1 SMP Wed Aug 29 11:51:58 MSK
```

```
2007
```

An oops usually starts with some description of what happened and ends with the Code string. Here is an example:

```
Aug 25 08:27:46 boar BUG: unable to handle kernel NULL pointer dereference at virtual
address 00000038
Aug 25 08:27:46 boar printing eip:
Aug 25 08:27:46 boar f0ce6507
Aug 25 08:27:46 boar *pde = 00003001
Aug 25 08:27:46 boar Oops: 0000 [#1]
Aug 25 08:27:46 boar SMP
Aug 25 08:27:46 boar last sysfs file:
Aug 25 08:27:46 boar Modules linked in: snapapi26(U) bridge(U) ip_vzredir(U) vzredir(U)
vzcompat(U) vzrst(U) i
p_nat(U) vzcpt(U) ip_conntrack(U) nfnetlink(U) vzfs(U) vzlinkdev(U) vzethdev(U)
vzevent(U) vzlist(U) vznet(U) vzstat(U) vzmo
n(U) xt_tcpudp(U) ip_vznetstat(U) vznetstat(U) iptable_mangle(U) iptable_filter(U)
ip_tables(U) vztable(U) vzdquota(U) vzdev(U) autofs4(U) hidp(U) rfcomm(U) l2cap(U)
bluetooth(U) sunrpc(U) ipv6(U) xt_length(U) ipt_ttl(U) xt_tcpmss(U) ipt_TCPMSS(U)
xt_multiport(U) xt_limit(U) ipt_tos(U) ipt_REJECT(U) x_tables(U) video(U) sbs(U)
i2c_ec(U) button(U) battery(U) asus_acpi(U) ac(U) lp(U) floppy(U) sg(U) pcspkr(U)
i2c_piix4(U) e100(U) parport_pc(U) i2c_core(U) parport(U) cpqphp(U) eepro100(U) mii(U)
serio_raw(U) ide_cd(U) cdrom(U) ahci(U) libata(U) dm_snapshot
(U) dm_zero(U) dm_mirror(U) dm_mod(U) megaraid(U) sym53c8xx(U) scsi_transport_spi(U)
sd_mod(U) scsi_mod(U) ext3(U) jbd(U) ehci_hcd(U) ohci_hcd(U) uhci_hcd(U)
Aug 25 08:27:46 boar CPU: 1, VCPU: -1.1
Aug 25 08:27:46 boar EIP: 0060:[<f0ce6507>] Tainted: P VLI
Aug 25 08:27:46 boar EFLAGS: 00010246 (2.6.18-028stab043.1-ent #1)
Aug 25 08:27:46 boar EIP is at clone_endio+0x29/0xc6 [dm_mod]
Aug 25 08:27:46 boar eax: 00000010   ebx: 00000001   ecx: 00000000   edx: 00000000
Aug 25 08:27:46 boar esi: 00000000   edi: b6f52920   ebp: c1a8dbc0   esp: 0b483e38
Aug 25 08:27:46 boar ds: 007b   es: 007b   ss: 0068
Aug 25 08:27:46 boar Process swapper (pid: 0, veid: 0, ti=0b482000 task=05e3f2b0
task.ti=0b482000)
Aug 25 08:27:46 boar Stack: 0b52caa0 00000001 00000000 b6f52920 00000000f0ce64de
00000000 02478825
Aug 25 08:27:46 boar 00000000 c18a8620 b6f52920 271e1a8c 024ca03800000000 00000000
00000000
Aug 25 08:27:46 boar 00000000 00000000 c18a3c00 00000202 c189e89400000006 00000000
05cb7200
Aug 25 08:27:46 boar Call Trace:
Aug 25 08:27:46 boar [<f0ce64de>] clone_endio+0x0/0xc6 [dm_mod]
Aug 25 08:27:46 boar [<02478825>] bio_endio+0x50/0x55
Aug 25 08:27:46 boar [<024ca038>] __end_that_request_first+0x185/0x47c
Aug 25 08:27:46 boar [<f0c711eb>] scsi_end_request+0x1a/0xa9 [scsi_mod]
Aug 25 08:27:46 boar [<02458f04>] mempool_free+0x5f/0x63
Aug 25 08:27:46 boar
Aug 25 08:27:46 boar [<f0c713c3>] scsi_io_completion+0x149/0x2f3 [scsi_mod]
Aug 25 08:27:46 boar [<f0c333b9>] sd_rw_intr+0x1f1/0x21b [sd_mod]
Aug 25 08:27:46 boar [<f0c6d3b9>] scsi_finish_command+0x73/0x77 [scsi_mod]
Aug 25 08:27:46 boar [<024cbfa2>] blk_done_softirq+0x4d/0x58
Aug 25 08:27:46 boar [<02426452>] __do_softirq+0x84/0x109
Aug 25 08:27:46 boar [<0242650d>] do_softirq+0x36/0x3a
Aug 25 08:27:46 boar [<024050b7>] do_IRQ+0xad/0xb6
Aug 25 08:27:46 boar [<024023fa>] default_idle+0x0/0x59
Aug 25 08:27:46 boar [<0240242b>] default_idle+0x31/0x59
Aug 25 08:27:46 boar [<024024b1>] cpu_idle+0x5e/0x74
Aug 25 08:27:46 boar =======================
Aug 25 08:27:46 boar Code: 5d c3 55 57 89 c7 56 89 ce 53 bb 01 00 00 00 83 ec 0c 8b 68
3c 83 7f 20 00 8b 45 00 8b 00 89 44 24 04 8b 45 04 89 04 24 8b 40 04 <8b> 40 28 89 44
24 08 0f 85 86 00 00 00 f6 47 10 01 75 0a 85 c9
```

```
Aug 25 08:27:46 boar EIP: [<f0ce6507>] clone_endio+0x29/0xc6 [dm_mod]
SS:ESP0068:0b483e38
Aug 25 08:27:46 boar Kernel panic - not syncing: Fatal exception in interrupt
```

All you need is to put the oops into a file and then send this file as part of your problem report to the Parallels support team.

Finding a Kernel Function That Caused the D Process State

If there are too many processes in the D state and you can't find out what is happening, issue the following command:

```
# objdump -Dr /boot/vmlinux-`uname -r` >/tmp/kernel.dump
```

and then get the process list:

```
# ps axfwln
  F UID   PID  PPID PRI NI  VSZ  RSS  WCHAN STAT TTY TIME COMMAND
100   0 20418 20417  17  0 2588  684      - R    ?   0:00 ps axfwln
100   0     1     0   8  0 1388  524 145186 S    ?   0:00 init
040   0  8670     1   9  0 1448  960 145186 S    ?   0:00 syslogd -m 0
040   0  8713     1  10  0 1616 1140 11ea02 S    ?   0:00 crond
```

Look for a number under the **WCHAN** column for the process in question. Then, open `/tmp/kernel.dump` in an editor, find that number in the first column and then scroll backward to the first function name, which can look like this:

```
"c011e910 <sys_nanosleep>:"
```

Then you can tell if the process “lives” or is blocked into the found function.

Problems with Container Management

This section includes recommendations on how to settle certain problems with Containers.

Failure to Start a Container

An attempt to start a Container fails.

Solution 1

If there is a message on the system console: `IP address is already used`, issue the `cat /proc/vz/veinfo` command. The information about the Container numeric identifier, Container class, number of Container’s processes and Container IP address shall be displayed for each running Container. This shall also demonstrate that your Container is up, i.e. it must be running without any IP address assigned. Set its IP address using the command:

```
# prlctl set <CT_ID> --ipadd <IP_address>
```

where *<CT_ID>* is the Container identifier and *<IP_address>* is the desired IP address.

Solution 2

Poor UBC parameters might prevent the Container from starting. Try to validate the Container configuration (see **Validating Container Configuration**). See what configuration parameters have caused the error and set appropriate values using the `prlctl set` command.

Solution 3

The Container might have used all its disk quota (disk space). Check the quota (see **Managing Disk Quotas** (p. 83) and **Chapter 4**) and adjust its parameters if needed (see **Setting Up Per-Container Disk Quota Parameters** (p. 85)).

Solution 4

Run the `prlctl console` utility to log in and get access to the Container console. The utility will provide Container startup/shutdown output that may be used to pinpoint the problem. For example:

```
# prlctl console 101
```

where 101 is a Container identifier.

Solution 5

Restore the latest working copy of the Container by means of the `prestore` utility (see **Managing virtual machine and Container Backups** (p. 37) for details).

Failure to Access a Container from Network

Solution 1

The IP address assigned to the Container might be already in use in your network. Make sure it is not. The problem Container address can be checked by issuing the following command:

```
# grep IP_ADDRESS /etc/vz/conf/<CT_ID>.conf
IP_ADDRESS="10.0.186.101"
```

The IP addresses of other Containers, which are running, can be checked by running

```
cat /proc/vz/veinfo
```

Solution 2

Make sure the routing to the Container is properly configured. Containers can use the default router for your network, or you may configure the server as rooter for its Containers.

Failure to Log In to a Container

The Container starts successfully, but you cannot log in.

Solution 1

You are trying to connect via SSH, but access is denied. Probably you have not set the password of the `root` user yet or there is no such user. In this case, use the `prlctl set --userpasswd` command. For example, for Container 101 you might issue the following command:

```
# prlctl set 101 --userpasswd root:secret
```

Solution 2

Check forwarding settings by issuing the following command:

```
# cat /proc/sys/ipv4/conf/venet0/forwarding
```

If it is 0 then change it to 1 by issuing the following command:

```
# echo 1 > /proc/sys/ipv4/conf/venet0/forwarding
```

Getting Technical Support

This section provides information on how to get technical support from Parallels.

Preparing and Sending Questions to Technical Support

In most cases, the support team must rely on the customer's observations and communications with the customer to diagnose and solve the problem. Therefore, the detailed problem report is extremely important. You can submit a support report by visiting the http://www.parallels.com/support/request web page and filling in the Online Support Form. When describing the problem, please do mention the following:

- symptoms of the problem
- when the problem began including the circumstances of the failure
- any changes you made to your system
- other information that may be relevant to your situation (e.g., the installation method)
- specific hardware devices that may be relevant to your problem

You can also use the Parallels Support Request Tracker tool to submit a problem report:

1. Follow the https://support.parallels.com link.
2. Enter the user name and password to log in to the Support Request Tracker.

   If you have not yet registered with the Support Request Tracker, click **Follow this link** and follow the instructions to get your login credentials.
3. Log in to the Support Request Tracker using the received credentials.
4. At the top of the displayed screen, select the name of your product and click the **New ticket in** button.
5. Follow the on-screen instructions to create a new support ticket.

Another way of getting help is to directly call us or visit one of our offices. The information about phone numbers, contact people and office addresses is available on the contact pages at http://www.parallels.com/contact.

Submitting a Problem Report to Technical Support

You can use the `prlctl problem-report` command to compile detailed reports for virtual machines and Containers experiencing problems and send them to the Parallels technical support. After receiving your report, the support team will closely examine your problem and make its best to solve it as quickly as possible.

Important: Reports contain only logs and the information on your system and network settings. They do not contain any private information.

To generate a report, specify the ID or name of the problem virtual machine or Container and the way to process the report:

- To automatically send the report to the Parallels technical support, pass the `-s` option to the command:

```
# prlctl problem-report <CT_ID|VM_name> -s
```

  This is the recommended way of running the command. If you use a proxy server to connect to the Internet, you need to additionally specify its parameters after the `--proxy` option:

```
# prlctl problem-report <CT_ID|VM_name> -s --proxy [user[:password]@]<proxyhost>[:port]
```

- To display the report on your screen in the machine-readable format, pass the `-d` option to the command. You can pipe the output to a file and then send it to the Parallels technical support, for example:

```
# prlctl problem-report <CT_ID|VM_name> -d > problemReport
```

  This command saves the generated report to the file `problemReport`.

Establishing Secure Channel to Parallels Support

Parallels Cloud Server provides you with a special tool - *Support Tunnel* - which allows you to establish a private secure channel to the Parallels support team server. After establishing such a channel, the support team will be able to quickly and securely connect to your Parallels server and diagnose and solve your problem. The secure connection to your server is achieved through a Virtual Private Network (VPN) created between the Parallels support team server and your server.

To start using the *Virtuozzo Support Tunnel* tool:

- Make sure the `openvpn` (version 2.0 and above) and `vzvpn` packages are installed on your server. Both packages are available in the Parallels Cloud Server remote repository, so you can install them using the `yum install` command:

```
# yum install openvpn
# yum install vzvpn
```

- Make sure that port 80 is opened on the server.

- Edit the `/etc/vzvpn/vzvpn.conf` file to specify the correct parameters for your proxy server, if you use any. For details on these parameters, see **vzvpn Configuration File** in the *Parallels Cloud Server 6.0 Command Line Reference Guide*.

After you have completed the tasks above and in case you encountered a problem, you can do the following to get assistance from the Parallels support:

1. Obtain a special certificate from Parallels which will uniquely identify you as a Parallels Cloud Server user. Certificates are issued by Parallels in the form of files and should be installed on your server by issuing the `vzvpn.sh key-install` *`certificate`* command where *`certificate`* denotes the name of the certificate file obtained from Parallels. You can get a certificate in one of the following ways:
   - Visit the http://www.parallels.com/en/support/virtuozzo/certificates web site, fill up the **Request Secure Virtuozzo Support Tunnel Certificate** form, and click the **Submit** button. After a while, a certificate will be generated and sent to the email address you provided in the **Request Secure Virtuozzo Support Tunnel Certificate** form.
   - Contact the Parallels support team via e-mail or by telephone and ask for a valid certificate.
2. After you are ready with the certificate installation, make sure your server is connected to the Internet.
3. On the server, execute the `/etc/init.d/vzvpn.sh start` command to establish a VPN between your server and the Parallels support server.
4. Contact the Parallels support team (by telephone or via e-mail) and inform them of the problem you encountered. You should also mention that you have launched the *Virtuozzo Support Tunnel* tool and established a VPN to the Parallels support server.
5. After that, the Parallels support team will connect to your server by using the secure VPN established, closely examine your problem, and make its best to solve the problem as quickly as possible.

Notes:

1. `Support Tunnel` is implemented as a standard Linux service running in the background of your system. Therefore, to have this service running after your server reboot, you should set it to the `autoboot` mode or start it manually again by executing the `/etc/init.d/vzvpn start` command.

2. To close the VPN session with the Parallels support server, you should issue the `/etc/init.d/vzvpn stop` command on the server.

CHAPTER 12

Glossary

This glossary defines terms and spells out abbreviations used in Parallels Cloud Server documentation. References to terms defined elsewhere in the glossary appear in italics.

Application template. A template used to install a set of applications in *Containers*. See also *Template*.

Container (or regular Container). A virtual private server, which is functionally identical to an isolated standalone server, with its own IP addresses, processes, files, users database, configuration files, applications, system libraries, and so on. Containers share one *Parallels server* and one OS kernel. However, they are isolated from each other. A Container is a kind of 'sandbox' for processes and users.

Guest operating system (Guest OS). An operating system installed inside a virtual machine and Container. It can be any of the supported Windows or Linux operating systems.

Hardware virtualization. A technology allowing you to virtualize physical servers at the hardware level. Hardware virtualization provides the necessary environment for creating and managing Parallels virtual machines.

Operating system virtualization (OS virtualization). A technology allowing you to virtualize physical servers at the operating system (kernel) level. OS virtualization provides the necessary environment for creating and managing Parallels Containers.

OS template (Operating System template). A template used to create new *Containers* with a pre-installed operating system. See also *Template*.

Package set. See *Template*.

Parallels Server. A hardware virtualization solution that enables you to efficiently use your physical server's hardware resources by sharing them between multiple virtual machines created on this server.

Parallels server (physical server or server). A server where the Parallels Cloud Server software is installed for hosting Parallels virtual machines and Containers. Sometimes, it is marked as Container 0.

Parallels Cloud Server license. A special license that you should install on the physical server to be able to start using Parallels Cloud Server. Every physical server must have its own license installed.

Memory and IOPS deduplication. A feature introduced in Parallels Cloud Server 6.0. By caching identical files in multiple Containers it helps eliminates memory and I/O bottlenecks and reduce the starting times of multiple similar Containers. Benefits Containers working in a Parallels Cloud Storage environment.

Parallels Virtuozzo Containers for Linux. An operating system virtualization solution allowing you to create multiple isolated Containers on a single physical server to share hardware, licenses, and management effort with maximum efficiency.

Private area. A part of the file system storing *Container* files that are not shared with other *Containers*.

Template (package set). A set of original application files (packages) repackaged for mounting. There are two types of templates. OS Templates are used to create new *Containers* with a pre-installed operating system. Application templates are used to install an application or a set of applications in *Containers*.

UBC. An abbreviation of *User Beancounter*.

User Beancounter. The subsystem of the Parallels Cloud Server software for managing *Container* memory and some system-related resources.

Virtual Environment (VE). An obsolete designation of a *Container*.

Virtuozzo File System (VZFS). A virtual file system for legacy Containers.

Virtual machine (VM). A computer emulated by Parallels Cloud Server. Like a Container, a virtual machine is functionally identical to an isolated standalone computer, with its own IP addresses, processes, files, users database, configuration files, applications, system libraries, and so on. However, unlike Containers, virtual machines run their own operating systems rather than sharing one operating system kernel.

Index

A

About This Guide - 11
Adding New Devices - 78
Advanced Tasks - 193
Aligning Disks and Partitions in Virtual Machines - 212
Applying New Configuration Sample to Container - 129
Assigning USB Devices to Virtual Machines - 84
Available Capabilities for Containers - 194

B

Backups Overview - 46
Basics of Hardware Virtualization - 21
Basics of OS Virtualization - 18
Binding CPUs to NUMA Nodes - 94

C

Capabilities Defined by POSIX Draft - 194
Changing System Time from Containers - 202
Changing Virtual Machine Disk Type - 101
Checking for Updates - 189
Choosing OS EZ Template - 32
Compacting Disks - 103
Configuring Additional Memory Parameters - 123
Configuring Capabilities - 194
Configuring CPU Affinity for Virtual Machines and Containers - 91
Configuring CPU Limits for virtual machines and Containers - 92
Configuring CPU Units - 91
Configuring Disk I/O Bandwidth - 112
Configuring IP Address Ranges for Host-Only Networks - 86
Configuring Legacy Containers - 120
Configuring Main Memory Parameters - 122
Configuring Main VSwap Parameters - 117
Configuring Network Bandwidth Management - 109
Configuring Network Classes - 105
Configuring Network Settings - 34
Configuring Offline Management - 171
Configuring OOM Killer Behavior - 119
Configuring Passwordless Access to the Source Node - 52
Configuring Priority Levels for Virtual Machines and Containers - 112
Configuring the Memory Allocation Limit - 118
Configuring the Number of I/O Operations Per Second - 113
Configuring veth Adapter Parameters - 159
Configuring Virtual Adapter Parameters - 162
Configuring Virtual Devices - 81
Configuring Virtual Machines and Containers in Bridged Mode - 154
Configuring Virtual Machines and Containers in Host-Routed Mode - 152
Configuring Virtual Network Parameters - 155
Connecting Containers to Virtual Networks - 160
Connecting Virtual Machines to Virtual Networks - 163
Connecting Virtual Networks to Adapters - 157
Connecting with a VNC Client - 206
Container Network Modes - 144
Converting Third-Party Virtual Machines and Disks - 86
Converting VZFS Containers to New Layout - 226
Copying Virtual Machines and Containers within Server - 39
CPU Limit Specifics - 94
Creating a Virtual Network - 154
Creating and Deleting veth Network Adapters - 158
Creating and Deleting Virtual Adapters - 161

Creating Configuration Files for New Linux Distributions - 211
Creating Customized Containers - 196
Creating Snapshots - 56
Creating Templates - 54
Creating Virtual Machines and Containers - 30
Creating VLAN Adapters - 143
Creating VZFS-based Containers - 225
Customizing Container Reinstallation - 74

D

Deleting Devices - 83
Deleting Snapshots - 59
Deleting Virtual Machines and Containers - 43
Deleting Virtual Networks - 158
Deploying Templates - 55
Determining Container Identifiers by Process IDs - 140
Differences Between Host-Routed and Bridged Network Modes - 151
Disabling Golden Image Functionality - 197
Disk Quota Parameters - 97
Documentation Conventions - 12

E

Enabling and Disabling Offline Management - 174
Enabling and Disabling Offline Services - 175
Enabling and Disabling Per-Container Quotas - 98
Enabling and Disabling Quotas - 99
Enabling CPU Hotplug for Virtual Machines - 95
Enabling Memory Hotplug for Virtual Machines - 126
Enabling VNC Access to Containers - 205
Enabling VNC Access to Virtual Machines - 205
Enabling VNC Access to Virtual Machines and Containers - 205
Enabling VPN for Containers - 76
Establishing Secure Channel to Parallels Support - 237

F

Failure to Access a Container from Network - 235
Failure to Log In to a Container - 235
Failure to Start a Container - 234
Feedback - 14
Finding a Kernel Function That Caused the D Process State - 233

G

General Considerations - 229
General Migration Requirements - 60
Getting Help - 13
Getting Technical Support - 236
Glossary - 239

H

Hardware Virtualization Layer - 21

I

Increasing Disk Capacity - 101
Initialize a Newly Added Disk - 79
Installing Optional Parallels Cloud Server Packages - 219
Installing the License - 179
Introduction - 10

K

Keeping Your System Up To Date - 186
Kernel Troubleshooting - 231

L

Learning Parallels Cloud Server 6.0 Basics - 15
Learning Private Networks - 165
Legacy Features - 223
License Statuses - 185
Linux-specific Capabilities - 195
Listing Adapters - 142
Listing Legacy Container Backups - 227
Listing Snapshots - 58
Listing Templates - 54
Listing Virtual Machines and Containers - 37
Listing Virtual Networks - 156
Locating Disk I/O Bottlenecks for Containers - 115

M

Main Operations on Services and Processes - 136
Making Screenshots - 84

Managing Adapters in Containers - 158
Managing Adapters in Virtual Machines - 161
Managing Container Resources Configuration - 127
Managing CPU Resources - 91
Managing Disk I/O Parameters - 112
Managing Disk Quotas - 97
Managing iptables Modules - 208
Managing Licenses - 178
Managing Memory Parameters for Containers - 117
Managing Memory Parameters for Virtual Machines - 122
Managing Network Accounting and Bandwidth - 105
Managing Network Adapters on the Parallels Server - 142
Managing Parallels Cloud Server Network - 141
Managing Per-Container Disk Quotas - 98
Managing Per-User and Per-Group Disk Quotas - 99
Managing Private Networks - 165
Managing Processes and Services - 137
Managing Resources - 89
Managing Services and Processes - 134
Managing Snapshots - 56
Managing Templates - 54
Managing Virtual Disks - 101
Managing Virtual Machine and Container Backups - 46
Managing Virtual Machine Configuration Samples - 130
Managing Virtual Machine Devices - 77
Managing Virtual Machine Disk Interfaces - 104
Managing Virtual Machines and Containers - 29
Managing Virtual Networks - 154
Memory and IOPS Deduplication - 19
Migrating Containers - 65
Migrating Containers from Servers with Parallels Virtuozzo Containers - 66
Migrating Containers to Virtual Machines - 65
Migrating Physical Computers to Virtual Machines and Containers - 67
Migrating Virtual Machines and Containers - 60
Migrating Virtual Machines and Containers Between Parallels Servers - 61
Migrating Virtual Machines to Containers - 71
Migrating Xen Virtual Machines - 71
Migration Restrictions for Containers - 69
Migration Restrictions for Virtual Machines - 71
Monitoring Processes in Real Time - 139
Monitoring Resources - 132
Mounting as a Specific User - 221

N

Network Traffic Parameters - 105
Networking Modes in Parallels Cloud Server - 144

O

Obtaining Server ID from Inside a Container - 203
Offline Management Configuration Files - 175
Organization of This Guide - 11
OS Virtualization Layer - 18

P

Parallels Cloud Server 6.0 Overview - 16
Parallels Cloud Server Configuration - 25
Parallels Containers - 18
Parallels Virtual Machines - 21
Pausing Virtual Machines - 77
Performing Container-Specific Operations - 74
Performing Initial Configuration - 34
Performing More Actions - 189
Performing Virtual Machine-Specific Operations - 77
Physical Server Availability Considerations - 28
Preparing and Sending Questions to Technical Support - 236
Problems with Container Management - 234

R

Rebootless Updates - 187
Reducing Disk Capacity - 102
Reinstalling Containers - 74
Requirements for Migrating to Containers - 68

Requirements for Migrating to Virtual Machines - 70
Resource Management - 26
Restarting Containers - 204
Reverting to Snapshots - 59
Running Commands in Virtual Machines and Containers - 42
Running Parallels Cloud Server 6.0 in Virtual Machines - 217

S

Saving Kernel Faults (OOPS) - 231
Scaling Container Configuration - 128
Setting Disk I/O Limits for Container Backups and Migrations - 115
Setting Immutable and Append Flags for Container Files and Directories - 207
Setting Passwords for Virtual Machines and Containers - 34
Setting Quota Parameters - 99
Setting Startup Parameters - 35
Setting Up Private Networks - 168
Sharing with Ownership of a Server User - 222
Sharing with Ownership of a Virtual Machine User - 220
Splitting Server Into Equal Pieces - 128
Standard Migration - 62
Starting, Stopping, and Querying Status of Virtual Machines and Containers - 36
Storing Extended Information on Virtual Machines and Containers - 38
Submitting a Problem Report to Technical Support - 236
Support of Virtual and Real Media - 23
Supported Guest Operating Systems - 31
Suspending Virtual Machines and Containers - 41

T

Templates - 20
Transferring the License to Another Server - 181
Troubleshooting - 228
Tuning VSwap - 120
Turning On and Off Network Bandwidth Management - 108

U

Understanding Licensing - 27
Understanding Offline Management - 171
Updating All Components - 187
Updating Containers - 191
Updating EZ Template Packages Inside a Container - 191
Updating EZ Templates - 188
Updating Kernel - 188
Updating OS EZ Template Caches - 192
Updating Parallels Cloud Server - 187
Updating Software In Virtual Machines - 190
Updating the Current License - 180
Using ALT+SYSRQ Keyboard Sequences - 231
Using Customized Application Templates - 200
Using Customized OS EZ Templates - 197
Using EZ OS Template Sets - 199
Using iptables Modules in Containers - 208
Using iptables Modules on Servers - 208
Using Open vSwitch Bridges - 176
Using OS Template Caches with Preinstalled Application Templates - 196
Using pbackup and prestore - 50
Using prlctl backup and prlctl restore - 47
Using Virtuozzo File System - 224

V

Viewing Active Processes and Services - 137
Viewing Detailed Information About Virtual Machines and Containers - 44
Viewing Disk I/O Statistics - 114
Viewing Network Traffic Statistics - 107
Viewing the Current License - 183
Viewing the License - 183
Virtual Machine Files - 23
Virtual Machine Hardware - 22
Virtual Machine Network Modes - 148

W

What are Disk Quotas? - 97
What are Resource Control Parameters? - 90
What Are Services and Processes - 135
Working with Shared Folders in Linux Virtual Machines - 220

Z

Zero-Downtime Migration - 63